AFIT/GE/ENG/97D-18

A GPS CODE TRACKING RECEIVER DESIGN FOR MULTIPATH MITIGATION USING MAXIMUM LIKELIHOOD ESTIMATION

THESIS
Fred Paul Baier
Captain, USAF

AFIT/GE/ENG/97D-18

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

AFIT/GE/ENG/97D-18

A GPS CODE TRACKING RECEIVER DESIGN FOR MULTIPATH MITIGATION USING MAXIMUM LIKELIHOOD ESTIMATION

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science

Fred Paul Baier, B.E.E.

Captain, USAF

December, 1997

AFIT/GE/ENG/97D-18

A GPS CODE TRACKING RECEIVER DESIGN FOR MULTIPATH MITIGATION USING MAXIMUM LIKELIHOOD ESTIMATION

Fred Paul Baier, B.E.E.

Captain, USAF

Approved:

Capt Stewart L. DeVilbiss, Ph.D. Thesis Advisor	3 Dec 97 Date
Dr. Meir Pachter Committee Member	Dec 3, 1997 Date
Maj Michael A. Temple, Ph.D. Committee Member	3 Dec 97 Date

Acknowledgements

First and foremost, I would like to thank my wife, Michele, for supporting me through the trying times of AFIT. She single-handidly ensured that our family was well taken care of while I was engulfed in my studies. My success at AFIT serves as a reflection of her successes in our home. I could not have done it without her love and the love of our children, Dalton and Ryan.

It is very easy to become engulfed in the trials of AFIT. Although I may forget the pain of the long stressful hours I endured over the eighteen month period, I will never forget the friendships that I have acquired along the way. It was a great priveledge to work with my GPS comrade, Chuck Ormsby. Thanks for all your help. I would also like to extend a special thankyou to Larken Hastriter and Vinod Naga who helped me produce my slides before my thesis presentation. I am eternally greatful.

I would also like to thank my advisor, Captain Stew DeVilbiss, and committee members, Dr. Meir Pachter and Major Mike Temple, for supporting my efforts through the very end. Their guidance and encouragement ensured that I achieved success.

My tenure at AFIT has been one of the most rewarding opportunities that I have ever experienced. The AFIT faculty and staff are among the best people for which I have ever had the opportunity to work with. My success at AFIT is largely due to the support extended by these people.

Long live AFIT!

Fred Paul Baier

Table of Contents

List of Figures

List of Tables

Abstract

The NAVSTAR Global Positioning System (GPS) is currently used in many applications requiring precise positioning data. Improving the precise positioning information requires the removal of errors that perturb the received signals. The errors introduced by multiple propagation channels, termed multipath, are not easily removed. These channels are caused by reflective surfaces near the receiver. As such, multipath is uncorrelated between receivers and, thus, cannot be removed through differencing techniques.

This thesis investigates a GPS code tracking loop design which uses maximum likelihood (ML) estimation to determine amplitude and phase information of the multipath signal which are used to adjust code tracking to account for multipath effects. Analysis of the operations that govern this design for the case of a single reflection shows that it has no steady state tracking error. Results of simulations indicate that the code tracking loop, in conjunction with the MLE, mitigate the effects of multipath and improves code tracking performance over the narrow correlator NCDLL for most scenarios analyzed. Overall results of simulations indicate that the implementation of the maximum likelihood estimator (MLE) in conjunction with the code tracking loop has the potential to enhance code tracking performance over that offered by the narrow correlator NCDLL in a GPS environment.

A GPS CODE TRACKING RECEIVER DESIGN FOR MULTIPATH MITIGATION USING MAXIMUM LIKELIHOOD ESTIMATION

I. Introduction

1.1 Overview

The NAVSTAR Global Positioning System (GPS) is currently used in many applications in both the military and civilian community to obtain precise positioning data. The popularity of GPS continues to grow, in part, because of its accessibility to both communities. Because of its wide use and acceptance, applications requiring increased precision continue to be discovered.

Improving the collection of precise ranging and positioning information requires the removal of errors that perturb the received signals. Many of the errors may be reduced or completely removed by differencing, a common technique used to remove measurement errors that are closely correlated between two or more nearby receivers. Differencing, however, does not reduce errors introduced by multiple propagation channels commonly known as multipath, because the multipath signals are unique for each receiver and thus uncorrelated.

The errors introduced as a result of multipath can be significant. As such, a considerable amount of research has been directed towards the mitigation of multipath. This mitigation, however, is complicated by the fact that the presence of multipath is not readily known. In addition, the effects of multipath vary with time, particularly in a dynamic environment such as that encountered by an aircraft in flight.

1.2 Problem

Multipath or multiple propagation paths occur when the transmitted GPS signal reflects off surrounding objects, such as the ground, buildings, the fuselage or wings of

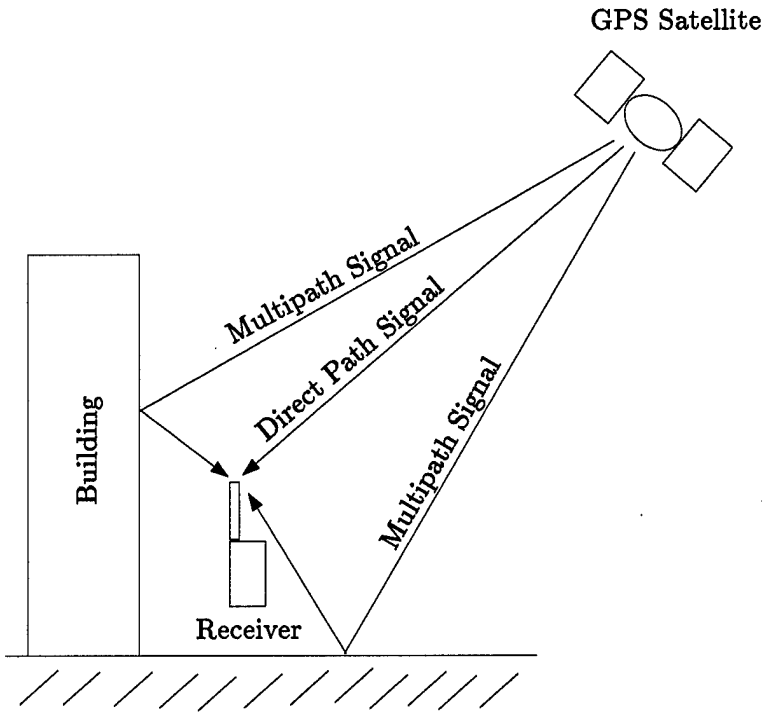


Figure 1 Multipath signal reflections

an aircraft, and mountains, prior to reception at the receiver as illustrated in Figure 1. The reflected signals experience a larger propagation delay relative to the direct path or line-of-sight (LOS) signal because of the additional distance traveled. Under certain conditions, these reflections induce errors in the code or phase synchronization process of the receiver that translate in turn from ranging errors into positioning errors. Because of the correlation characteristics of the spreading code, for traditional code tracking loops, multipath signals with delays greater than $1.5T_c$, where T_c is the chip period of the spreading code, have no impact on the pseudo-range measurement [19]. This delay interval equates to signals that are reflected within 450 meters of the receiver. However, multipath signals within 450 meters (delays less than $1.5T_c$) can introduce significant errors into the ranging measurements. This thesis investigates a GPS code tracking receiver which uses maximum likelihood (ML) estimation to mitigate the effects of multipath.

1.3 Summary of Current Knowledge

To investigate the performance of code tracking loops in the presence of multipath, one must understand the composition of the transmitted signal and the multipath composition of the received signal. The following section discusses these issues and also describes the correlation operation required of GPS receivers.

1.3.1 The GPS Signal. GPS signals are composed of three basic elements: the wide band spreading code, the carrier signal, and the navigation message or navigation code. The wide band spreading code possesses pseudo random properties which essentially appear as noise within the signal bandwidth. Upon successful correlation with a locally generated replica, the spreading code provides a processing gain which allows signals to be transmitted near or below the noise floor.[1] GPS incorporates two types of spreading codes: course/acquisition (C/A) code and precision code (P-code). C/A code is accessible to the civilian community while P-code is intended for military use only. The C/A code has a chip rate or bit rate of $1/T_c = 1.023\ MHz$ and a code period, $N = 1023$ chips; while the P-code has a chip rate, $1/T_c = 10.23\ MHz$ and a code period, $N \approx$ one week. The null to null bandwidth of each code is 2.046 MHz and 20.46 MHz, respectively. The GPS satellite transmits right hand circularly polarized (RHCP) carrier signals within the L-band at two carrier frequencies, $L1 = 1575.42\ MHz$ and $L2 = 1227.6\ MHz$. The C/A code and P-code are both mixed with the L1 carrier while only the P-code is transmitted via the L2 carrier. From this point forward, it is assumed that only C/A code is being tracked unless otherwise stated.

The navigation message provides satellite ephemerides, system time, correction data, satellite clock behavior data, and status messages, necessary to ensure correct positioning solutions. This data message has a bit rate of 50 $bits/sec$. The navigation data is combined with the pseudo random noise (PRN) ranging code or spreading code by Modulo-2 addition and can be represented as a string of ones and zeros. This bit stream is then modulated onto the carrier using binary phase shift keying (BPSK), which results in instantaneous

[1]The processing gain, which is the ratio between the spreading code chip rate and the data modulation bit rate, for GPS C/A code is $\approx 2e4 = 43dB$

phase changes of the carrier by 180 degrees. This bandwidth spreading process is known as direct-sequence spread spectrum (DS/SS). The null-to-null bandwidth of the transmitted signal is 2.046 MHz, the bandwidth of the spreading code, and is centered about the L1 carrier frequency. The transmitted signal can be represented as

$$s(t) = \sqrt{2P} m(t)\, c(t) \cos(2\pi f_o t + \theta) \tag{1}$$

where P is the power of the signal, $c(t)$ is the spreading code, $m(t)$ is the navigation message, f_o is the carrier frequency in Hz, and θ is the transmitted signal phase.

1.3.2 Multipath Propagation. The influence of the reflected signals depends largely on their amplitudes and delay relative to the direct path signal. The direct signal will have more power than the reflected signals, unless there are obstructions in the LOS that reduce the power level of the direct signal. This is attributed, in part, to the longer distance that the reflected signals must travel, which causes additional free space loss. More significantly, there is typically attenuation caused by the reflector. Finally, the antenna may cause some attenuation due to the gain pattern associated with an orthogonal polarization. A reflected signal will typically reverse to left hand circular polarized (LHCP) while the LOS signal will be RHCP.

The received GPS signal in the presence of multipath can be expressed as

$$\begin{aligned} r_{mp}(t) &= \sqrt{2P} a_0 m(t-\tau_o) c(t-\tau_o) cos(2\pi f_o t + \theta_0) \\ &\quad + \sqrt{2P} \sum_{i=1}^{M-1} a_i m(t-\tau_o-\alpha_i T_c) c(t-\tau_o-\alpha_i T_c) cos(2\pi f_o t + \theta_i) + n(t) \end{aligned} \tag{2}$$

where P is the power of the signal, M represents the number of signals received, a_i is the attenuation coefficient of the i^{th} signal, $c(t)$ is the spreading code, $m(t)$ is the navigation message, f_o is the carrier frequency in Hz, τ_o is the propagation delay of the direct path signal, $\alpha_i T_c$ is the propagation delay of the i^{th} signal from (relative to) the direct path signal, θ_i is the phase of the i^{th} signal, and $n(t)$ is the received noise signal. The noise

term can be expressed as [9]

$$n(t) = \sqrt{2}\left[n_I(t)\cos(2\pi f_o t) - n_Q(t)\sin(2\pi f_o t)\right] \tag{3}$$

where $n_I(t)$ and $n_Q(t)$ are the in-phase and quadrature noise components, respectively.

For the case where the direct path signal and one reflected signal is received, the signal can be expressed as

$$\begin{aligned} r_{mp}(t) &= \sqrt{2P}a_0 m(t-\tau_o)c(t-\tau_o)cos(2\pi f_o t + \theta_0) \\ &\quad +\sqrt{2P}a_1 m(t-\tau_o-\alpha T_c)c(t-\tau_o-\alpha T_c)cos(2\pi f_o t + \theta_1) + n(t) \end{aligned} \tag{4}$$

where αT_c ($\alpha = \alpha_1$ for Equation 2) represents the delay of the reflected signal relative to the direct path signal. The average power-delay for the direct path signal and one reflected signal can be modeled as $Pa_0^2\delta(t-\tau_o) + Pa_1^2\delta(t-\tau_o-\alpha T_c)$ as shown in Figure 2.

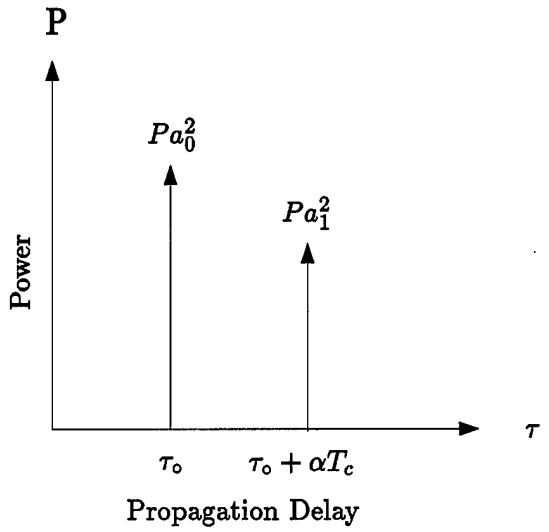


Figure 2 Average power-delay for direct signal and one reflection

As stated earlier, for traditional code tracking loops, multipath signals with delays less than $1.5T_c$ can introduce significant errors into the ranging measurements. For example, a code phase tracking error, ϵ, of one-tenth of a chip, $0.1T_c$, results in a tracking

difference of

$$\epsilon = 0.1T_c \approx 98ns \tag{5}$$

which equates to a ranging error of

$$\begin{aligned} PR_e &= (3e8\ m/s)(9.8e9\ m/s) \\ &= 29.4\ m. \end{aligned} \tag{6}$$

1.3.3 Receiver Operation. All GPS receivers incorporate some sort of correlation process in their design in order to remove the direct sequence spread spectrum (DS/SS) code from the signal (i.e.,"despread" the signal). This receiver despreading operation requires a correlation of the received signal with a locally generated spreading code replica, which uniquely corresponds to the desired signal. Conceptually, a received signal that has been spread using a different spreading code, corresponding to a different transmitter, will appear as noise and will cause minimal interference with the desired signal; this interference corresponds to the cross-correlation between the two spreading codes, which can be expressed as

$$R_c(\tau) = 1/NT_c \int_0^{NT_c} c'(t)\, c(t+\tau)\, dt \tag{7}$$

where N represents the number of chips per code cycle and both waveforms have the same chip period T_c.

When $c(t) = c'(t)$, a sufficient approximation of the PRN code cross-correlation function is given by the following:

$$R_c(\tau) = \begin{cases} 1 - \frac{|\tau|}{T_c} & \text{for } |\tau| \leq T_c \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Observe that maximum correlation occurs when $\tau = 0$, corresponding to [perfect synchronization. Figure 3 illustrates the correlation function approximation given in Equation 8. Note that the peak of the function represents maximum correlation between the locally

generated spreading code and the received signal. As the two signals separate, the correlation function approaches zero, indicating the two signals are no longer correlated. As such, the received signal can no longer be detected in the integration of Equation 7.

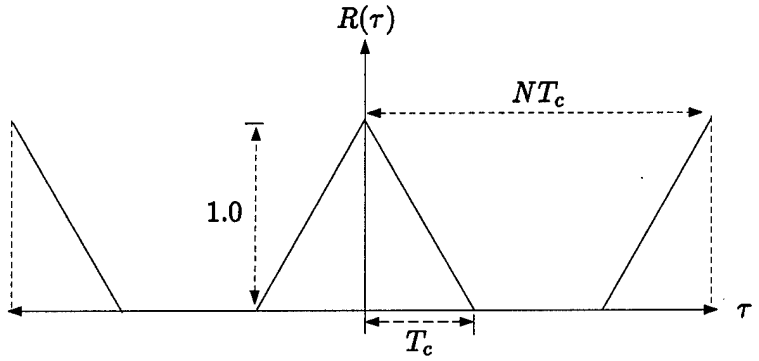


Figure 3 Direct sequence/spread spectrum (DS/SS) spreading code correlation function.

1.4 Previous Research

Currently, errors in receivers have been reduced to near theoretical limits for receivers processing ideal GPS signals. However, non-ideal signals possess characteristics that reduce positioning accuracy. These errors include selective availability (SA)[2], satellite clock bias, atmospheric effects, ephemeris errors, and multipath. As stated previously, the influence of most of these errors can be greatly reduced or removed entirely through differencing techniques. Such techniques do not effectively reduce multipath errors because of the lack of multipath correlation between receivers.

Numerous techniques have been investigated in the attempt to reduce the effects of multipath. Several techniques attempt to take advantage of geometrical differences between the direct and the reflected signals. Other techniques involve modifying and creating different receiver designs by enhancing signal processing techniques. This thesis limits its analysis to techniques associated with the latter.

The delay-lock loop (DLL) is the most common code tracking loop design used in GPS receivers. It is composed of two correlators using a relative delay spacing of ΔT_c

[2]The intentional degradation of the satellite transmitted signal is termed selective availability.

between each other. The 'early' correlator mixes a code replica with the received signal that is advanced by $\Delta T_c/2$ from the received signal. The 'late' correlator mixes a code replica with the received signal that is delayed by $\Delta T_c/2$ from the received signal. The signals in each of the correlator branches are processed to form a tracking error signal. Depending on how the correlated signals are processed, the DLL can be classified as being either coherent or noncoherent. The coherent DLL requires carrier phase information to process the received signals whereas the noncoherent DLL (NCDLL) operates independent of carrier phase tracking. Traditionally, receivers employing the NCDLL have used a normalized correlator spacing, $\Delta = 1.0$.[3] It has been determined, however, that narrowing the correlator spacing improves tracking performance [8,18].

Various designs have been derived from the basic properties of the NCDLL. However, new designs typically include estimation hardware to determine the amplitude, phase, and delay of the multipath components of a received signal. Two such designs are the Multipath Estimating DLL (MEDLL) and the Modified RAKE DLL (MRDLL).

MEDLL estimates the amplitude, delay, and phase of each multipath component using ML criteria. The estimates are used to remove the interfering correlation functions of the reflected signals. Thus, only the direct-path correlation function is tracked [14,16,17].

MRDLL also uses ML criteria to estimate the multipath parameters. These estimates are provided to a multiple correlator tracking loop. Although the parameters of multiple reflected signals can be estimated, MRDLL has only been simulated under the conditions with one reflected signal present [6,7].

1.5 Scope

This research investigates the use of ML estimation to improve code tracking performance. MRDLL serves as the platform in which the ML estimator (MLE), called the multiple correlator estimation unit (MCEU), is implemented and tested. In addition, the tracking performance of MRDLL is compared with the tracking performance of the NCDLL (with a normalized correlator spacing, $\Delta = 1.0$ and $\Delta = 0.1$).

[3] 'Normalized' implies a spacing Δ relative to one chip.

1.6 Assumptions

The following assumptions are made in the performance of this research:

1. The received GPS signal consists of a direct-path signal and one multipath signal.[4]
2. The code tracking loop has acquired the phase of the incoming PN sequence.
3. Only C/A code is being tracked.
4. The received GPS signal corresponds to only one GPS satellite.
5. Only Signal processing techniques are considered for multipath mitigation.
6. BPSK data modulation on the received signal is not addressed.
7. Doppler effects on code tracking performance are considered negligible.

1.7 Approach Methodology

An in-depth analysis of the MCEU is performed under ideal theoretical conditions. An enhanced model of MRDLL (referred to from here on as eMRDLL), which implements the MCEU, is provided. Using the model, further analysis is conducted in a simulated code tracking environment under realistic multipath conditions. In addition, the tracking performance of eMRDLL in the presence of additive white Gaussian noise (AWGN) environment is examined. A model of the NCDLL is provided to investigate the tracking performance of the NCDLL (to include narrow correlator spacing) versus the tracking performance of eMRDLL in a simulated multipath environment.

1.8 Materials and Equipment

Theoretical models of the MCEU were implemented using Matlab, version 4.2c, and the Optimization Toolbox from The MathWorks, Inc. of Natick, Massachusetts. Simulations of the multipath environment and models of eMRDLL and the NCDLL were developed using the Simulink Toolbox, version 1.3, the Signal Processing Toolbox, and the Communications Toolbox which are also from The MathWorks, Inc. Simulations were performed on the Sun Workstations provided at the Air Force Institute of Technology (AFIT).

[4]Note: The MLE implemented is designed to estimate the parameters of multiple reflected signals.

II. Background

2.1 Overview

As stated in Chapter 1, numerous techniques have been applied in the attempt to reduce the effects of multipath. Some techniques try to take advantage of the geometry of the direct signal and the reflected signals, by varying antenna design and/or placement. Other techniques involve modifying and creating different receiver designs. This chapter examines receiver designs that incorporate signal processing techniques to mitigate the effects of multipath. Improved signal processing techniques in receiver designs offer the greatest potential for mitigating the effects of multipath in all applications, particularly with the recent advances in technology.

2.2 Delay Lock Loop

One of the predominantly used code tracking loop configurations is the delay lock loop (DLL). Commonly referred to as the 'early-late' delay lock loop, this configuration can be designed as a coherent or noncoherent DLL (NCDLL) depending on the desired application. The coherent DLL requires accurate carrier phase tracking whereas the NCDLL can operate without carrier aiding. As such, the NCDLL is less likely to loose lock due to cycle slips of the carrier, thus making it more robust.

Because of its robustness, the NCDLL is employed in many receiver designs to perform code loop tracking. It is composed of two correlators with a spacing of the duration of ΔT_c as shown in Figure 4. The PN generator provides the 'early' gate with a code replica that is advanced by $\Delta T_c/2$ and provides the 'late' gate with a code replica that is delayed by $\Delta T_c/2$ relative to the received signal. Prior to narrow correlator spacing advancements, a correlator spacing of T_c, where $\Delta = 1.0$, was typically used. The correlator outputs of each channel are bandpass filtered and then squared and low pass filtered to remove the effects of data modulation and carrier phase shift. The outputs of the two channels are then differenced to generate the code phase error signal, $\epsilon(t, \delta)$, which is subsequently filtered; this signal is used to drive the voltage control clock (VCC) which corrects the code phase error of the local PN code generator (i.e., provides feedback).

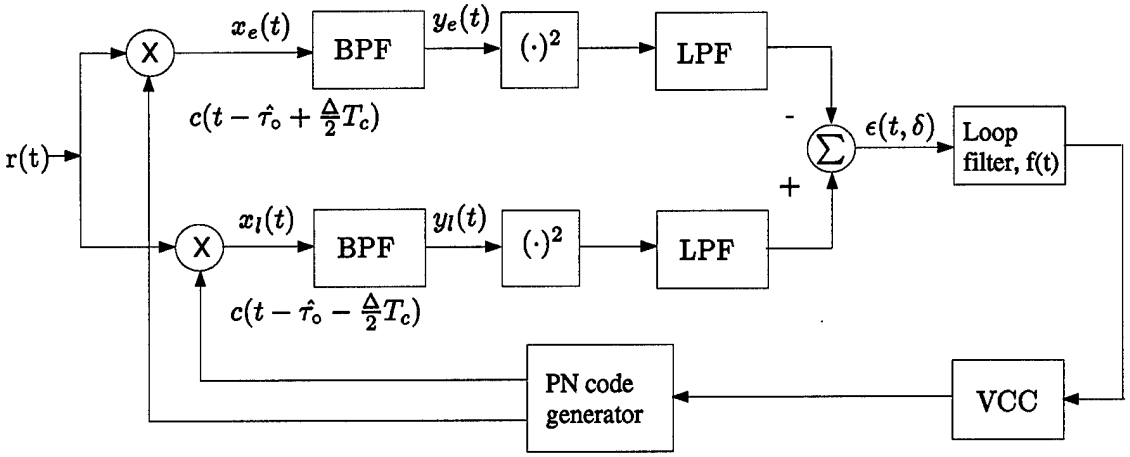


Figure 4 Non-coherent Delay Lock Loop.

2.2.1 Analysis of the NCDLL. The received signal entering the NCDLL, neglecting the navigation message and multipath, can be expressed as

$$r(t) = \sqrt{2P}a_0c(t-\tau_o)cos(2\pi f_o t + \theta) + n(t) \tag{9}$$

where $n(t)$ is the noise term presented in Equation 3. For this analysis, the navigation message can be neglected because this results in the maximum possible noise component in the power spectral density of the discriminator, S_ϵ, within the tracking loop bandwidth. This can be verified by realizing that without data modulation, the PSD of the data, $S_{\theta_d}(f)$, is a delta dirac function at $f = 0$ and the convolution process does no spectral widening of the (signal×noise) term [9]. The received signal enters into the 'early' and 'late' branches of the NCDLL where it is mixed with a locally generated replica of the spreading code represented by $c(t-\hat{\tau}_o \pm \Delta T_c/2)$ in which the + sign indicates the early and − sign the late branch and $\hat{\tau}_o$ represents the estimate of τ_o. The resulting signal(s) of the 'early' and 'late' branches are

$$x_e(t) = \sqrt{2P}a_0c(t-\tau_o)c(t-\hat{\tau}_o+\Delta T_c/2)cos(2\pi f_o t+\phi) + n(t) \tag{10}$$

and,

$$x_l(t) = \sqrt{2P}a_0c(t-\tau_o)c(t-\hat{\tau}_o-\Delta T_c/2)cos(2\pi f_o t+\phi) + n(t). \tag{11}$$

The mixed signals in each channel pass through a bandpass filter (BPF) centered at f_o Hz. The BPF characteristics should be chosen such that it passes only the carrier frequency plus or minus any Doppler offset. Simon [11] has shown that code tracking performance may be improved, particularly in deep noise, by selecting the filter bandwidth on the order of the data rate; however, Doppler offset must still be taken into consideration.

For systems with high spread spectrum processing gain, such as GPS, the components of interest are

$$y_e(t) = \sqrt{2P}a_0 c(t-\tau_o)c(t-\hat{\tau}_o + \Delta T_c/2)cos(2\pi f_o t + \phi) + n_e(t) \tag{12}$$

and

$$y_l(t) = \sqrt{2P}a_0 c(t-\tau_o)c(t-\hat{\tau}_o - \Delta T_c/2)cos(2\pi f_o t + \phi) + n_l(t) \tag{13}$$

where the noise components of the early and late channels are given by

$$n_e(t) = \left[c(t-\hat{\tau}_o + \frac{\Delta}{2}T_c)n(t)\right]_{BPF} \tag{14}$$

and

$$n_l(t) = \left[c(t-\hat{\tau}_o - \frac{\Delta}{2}T_c)n(t)\right]_{BPF} \tag{15}$$

respectively. Although code self noise components are present as a result of the mixing operation, these components can be neglected for systems with high processing gains [9] and where the single-sided loop bandwidth B_L[1] is much less than the PN code chip rate (1.023 MHz for GPS C/A code) [11].

Considering that the dc component of the spreading waveform product is the auto-correlation function $R_c(\tau)$ of the spreading waveform evaluated at $\tau = \tau_o - \hat{\tau_o} \pm \Delta T_c/2$,

[1]GPS loop bandwidths range from 0.02 to 1 Hz [19]

the average 'early' and 'late' channel signals can be represented as

$$\overline{y_e}(t) = \sqrt{P}a_0 R_c[(\delta + \Delta/2)T_c]cos(2\pi f_o t + \phi) + \overline{n_e}(t) \tag{16}$$

and

$$\overline{y_l}(t) = \sqrt{P}a_0 R_c[(\delta - \Delta/2)T_c]cos(2\pi f_o t + \phi) + \overline{n_l}(t) \tag{17}$$

where $\delta = (\tau - \hat{\tau_o})/T_c$ is the normalized code phase estimation error and $\overline{n_e}(t)$, $\overline{n_l}(t)$ denote time averaged noises. The signal in each channel passes through a square law device and then is low pass filtered. The LPF removes the high frequency term, $4\pi f_o$, caused by the squaring operation in each channel. The difference of the two channels provides the tracking discriminator/error signal, $\epsilon(t,\delta) = [y_l^2]_{LPF} - [y_e^2]_{LPF}$ given as

$$\epsilon(t,\delta) = Pa_0^2 S_\Delta(\delta) + n_\epsilon(t). \tag{18}$$

where $S_\Delta(\delta)$ is the delay lock loop discriminator (referred to as the S-curve) defined as

$$S_\Delta(\delta) \triangleq R_c^2\left[\left(\delta - \frac{\Delta}{2}\right)T_c\right] - R_c^2\left[\left(\delta + \frac{\Delta}{2}\right)T_c\right] \tag{19}$$

and $n_\epsilon(t)$ represents the noise component of $\epsilon(t,\delta)$. The plot of the tracking error for varying values τ forms a tracking curve. Different values of 'early-late' gate spacing provide different curve characteristics, such as a change in slope of the tracking curve which in turn causes a change in the effective tracking range. The tracking curve is linear for the normalized correlator spacing $\Delta \leq 1.0$; however, it becomes non-linear for values of $\Delta >$ 1.0. Therefore, correlator spacings greater than 1.0 chips are not typically implemented.

The delay lock discriminator for $\Delta \leq 1.0$ can be characterized as [9]

$$S_\Delta(\delta) = \begin{cases} 0 & \text{for } -N+1+\frac{\Delta}{2} < \delta < -(1+\frac{\Delta}{2}) \\ \frac{1}{N^2} - [1+(\delta+\frac{\Delta}{2})(1+\frac{1}{N})]^2 & \text{for } -(1+\frac{\Delta}{2}) < \delta < (\frac{\Delta}{2}-1) \\ -2(1+\frac{1}{N})\Delta[1+(1+\frac{1}{N})\delta] & \text{for } (\frac{\Delta}{2}-1) < \delta < -\frac{\Delta}{2} \\ 2(1+\frac{1}{N})[2-(1+\frac{1}{N})\Delta]\delta & \text{for } -\frac{\Delta}{2} < \delta < +\frac{\Delta}{2} \\ 2(1+\frac{1}{N})\Delta[1-(1+\frac{1}{N})\delta] & \text{for } \frac{\Delta}{2} < \delta < (1-\frac{\Delta}{2}) \\ \left[1-(1+\frac{1}{N})(\delta-\frac{\Delta}{2})\right]^2 - \frac{1}{N^2} & \text{for } (1-\frac{\Delta}{2}) < \delta < (1+\frac{\Delta}{2}) \end{cases} \tag{20}$$

For the traditional value of $\Delta = 1.0$, the resulting tracking curve is presented in Figure 5.

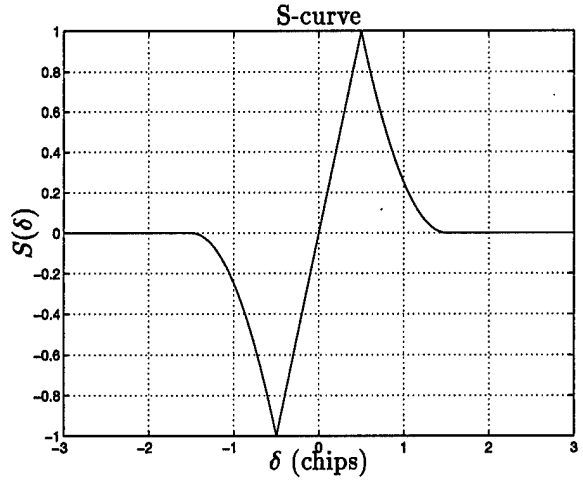


Figure 5 S-curve for NCDLL, $\Delta = 1$

2.2.2 Determination of Noise, $n_\epsilon(t)$. The noise components, $n_e(t)$ and $n_l(t)$, at the output of the BPF are the products of a bandlimited white Gaussian noise process $n(t)$ centered at f_o and the spreading code waveforms $c(t-\hat{\tau}_o \pm \frac{\Delta}{2}T_c)$ as described in Equations 14 and 15. This process provides the benefit of spreading the noise via the spreading code waveforms and thereby reducing the power spectral density in the frequency range passed by the BPF. The power spectrum of $n(t)$ after passing through the BPF can be expressed as

$$S_{n':e,l}(f) = S_{n:e,l}(f)|H(f)|^2. \tag{21}$$

Since the power spectrum of $n(t)$ is much wider in bandwidth than the power spectrum of $c(t)$, convolution has little effect. Assuming ideal filters, the output PSD of the noise component, $S_{n':e,l}(f)$, has a magnitude of $\frac{N_0}{2}$ and two sided bandwidth B_N and is centered about f_o.

The noise at the discriminator output, $n_\epsilon(t)$, is a function of $\epsilon(t,\delta)_{sig} = [y_l^2]_{LPF} - [y_e^2]_{LPF}$. As stated earlier, when the received carrier is not modulated with data, the maximum possible noise component is produced in the power spectral density, $S_{n_\epsilon}f)$, within the tracking loop bandwidth [9]. Because the loop filter bandwidth, B_L, following the discriminator is much smaller than $n_\epsilon(t)$, the dc component of $S_{n_\epsilon}(f)$ provides an accurate approximation of the noise PSD. This PSD is approximated as [4, 9]

$$\begin{aligned} S_{n_\epsilon}(f) \approx S_{n_\epsilon}(f)|_{f=0} &= 2N_0^2 B_N \\ &\quad +2Pa_0^2 N_0 \left\{ R_c^2 \left[\left(\delta - \frac{\Delta}{2} \right) T_c \right] - R_c^2 \left[\left(\delta + \frac{\Delta}{2} \right) T_c \right] \right\} \end{aligned} \quad (22)$$

over the loop bandwidth (zero elsewhere) and has units of W/Hz.

2.2.3 NCDLL Linear Model. The delay lock discriminator is determined by a non-linear process (because of the squaring operation in the square law device). However, a linear model, which will be developed in this section, can still be used to describe the DLL.

Ultimately, the code phase tracking error is determined by the difference of the phase of the received signal and the code phase, produced by the VCC, and is given as

$$\begin{aligned} r_d(t) &= K_d \delta \\ &= K_d \left(\frac{\tau_o - \hat{\tau}_o}{T_c} \right) \end{aligned} \quad (23)$$

where δ represents the normalized code phase error, τ_o/T_c represents the normalized code phase of the received signal, $\hat{\tau}_o/T_c$ represents the normalized estimated code phase from the VCC, and K_d represents the gain of the mixing process in *volts*. The VCC operates

at an angular frequency represented as

$$2\pi f_{VCC}(t) = 2\pi f_q + K_o r_f(t) \tag{24}$$

where $2\pi f_q$ is the center frequency (rad) of the VCC, K_o is the VCC gain in ($rad/sec\ volts$), and $r_f(t)$ is the output signal of the loop filter [1]. When $\delta = 0$, the VCO is operating at the center or quiescent frequency, $2\pi f_q$. If the input signal is positive at a specific instant in time, the phase of the input signal then starts leading the phase of the output signal of the VCC. An error signal, $\epsilon(t, \delta)$, develops at the discriminator output which increases with time. With a delay due to the loop filter, $r_f(t)$ will increase with time causing the VCC to increase its frequency until the VCC's frequency matches the input's signal's frequency. The VCC then operates at a new frequency equal to $2\pi(f_o + \Delta f)$. The signal at the output of the loop filter, $r_f(t)$, will settle to a final value of $r_f = 2\pi\Delta f/K_o$.

The angular frequency of a signal is defined as the first derivative of its phase with respect to time, $2\pi f = d\theta/dt$. Hence, the phase of a signal is the time integral of its angular frequency. Equation 24 integrated over time is [9]

$$2\pi\frac{\hat{\tau_o}}{T_c}(t) = 2\pi K_o \int_0^t r_f(\alpha)d\alpha \tag{25}$$

which simplifies to

$$\frac{\hat{\tau_o}(t)}{T_c} = K_o \int_0^t r_f(\alpha)d\alpha. \tag{26}$$

where $\hat{\tau_o}(t)/T_c$ represents the phase of the VCC. The output of the loop filter can be described by its impulse response $f(t)$ convolved with the input signal as follows

$$\begin{aligned} r_f(\alpha) &= \epsilon(t, \delta) * f(\lambda)d\lambda \\ &= \int_{-\infty}^{\alpha} \epsilon(\alpha, \delta) f(\alpha - \lambda)d\lambda. \end{aligned} \tag{27}$$

Incorporating Equation 27 into Equation 26, the output phase of the VCC can be expressed as

$$\frac{\hat{\tau}_o(t)}{T_c} = K_o \int_0^t \int_{-\infty}^{\alpha} \epsilon(\alpha, \delta) f(\alpha - \lambda) d\lambda d\alpha. \tag{28}$$

Substituting Equation 18 into Equation 28 provides the equation for the nonlinear model

$$\frac{\hat{\tau}_o(t)}{T_c} = K_o \int_0^t \int_{-\infty}^{\alpha} \left(P a_0^2 S_\Delta(\delta) + n_\epsilon(\lambda) \right) f(\alpha - \lambda) d\lambda d\alpha. \tag{29}$$

For small code tracking errors, $-\frac{\Delta}{2} < \delta < +\frac{\Delta}{2}$, Equation 20 provides

$$\begin{aligned} S_\Delta(\delta) &= 2\left(1 + \frac{1}{N}\right)\left[2 - \left(1 + \frac{1}{N}\right)\Delta\right]\delta \\ &= 4\left(1 + \frac{1}{N}\right)\left[1 - \left(1 + \frac{1}{N}\right)\frac{\Delta}{2}\right]\delta. \end{aligned} \tag{30}$$

To form a linear model, shown in Figure 6, the noise term, $n_\epsilon(t)$, of the discriminator is moved to the loop input with the appropriate gain adjustment provided as follows

$$K_d = 4\left(1 + \frac{1}{N}\right)\left[1 - \left(1 + \frac{1}{N}\right)\frac{\Delta}{2}\right]\left(P a_0^2\right). \tag{31}$$

Based on the linear model shown in Figure 6, the Laplace transform of the tracking loop

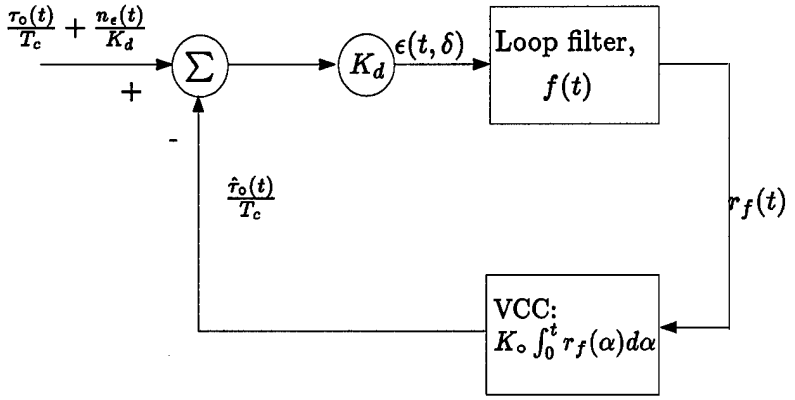


Figure 6 NCDLL linear equivalent circuit.

output $\hat{\tau}_o$ relative to the loop's input, τ_o, can be represented via

$$\frac{\hat{\tau}_o(s)}{T_c} = \frac{\tau_o(s) - \hat{\tau}_o(s)}{T_c}\left[K_d K_o \frac{F(s)}{s}\right]. \tag{32}$$

Solving for $\frac{\hat{\tau}_o(s)}{\tau_o(s)}$ given Equation 32 results in the closed loop transfer function

$$H(s) \triangleq \frac{\hat{\tau}_o(s)}{\tau_o(s)} = \frac{K_d K_o F(s)}{s + K_d K_o F(s)} \tag{33}$$

which characterizes the DLL. Commonly, a simple first order lead-lag filter is implemented as the loop-filter, $F(s)$. The transfer function of the loop filter can be expressed as

$$F(s) = \frac{1 + \tau_2 s}{\alpha_1 + \tau_1 s} \tag{34}$$

where [3]

$$\alpha_1 = \begin{cases} 1 & \text{passive filter} \\ 0 & \text{active filter} \end{cases} \tag{35}$$

It has been determined that better performance will almost always be obtained through the use of an active filter [2]. Assuming the use of a first-order active filter, produces a second order loop transfer function

$$\begin{aligned} H(s) &= \frac{K_d K_o \left(\frac{1+\tau_2 s}{\tau_1 s}\right)}{s + K_d K_o \left(\frac{1+\tau_2 s}{\tau_1 s}\right)} \\ &= \frac{\frac{K_d K_o}{\tau_1 s} + \frac{K_d K_o \tau_2}{\tau_1}}{s + \frac{K_d K_o \tau_2}{\tau_1 s} + \frac{K_d K_o \tau_2}{\tau_1}} \\ &= \frac{\left(\frac{K_d K_o \tau_2}{\tau_1}\right) s + \frac{K_d K_o}{\tau_1}}{s^2 + \left(\frac{K_d K_o \tau_2}{\tau_1}\right) s + \frac{K_d K_o}{\tau_1}} \end{aligned} \tag{36}$$

Equation 36 can then be represented in the classical manner for servo mechanisms as [2,9]

$$H(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{37}$$

where the loop natural frequency, ω_n in $\frac{rad}{sec}$, and the damping factor, ζ (unitless), are defined as

$$\omega_n = \sqrt{\frac{K_d K_o}{\tau_1}} \tag{38}$$

and

$$\zeta = \frac{\tau_2}{2}\omega_n \tag{39}$$

respectively. The single-sided noise equivalent bandwidth, B_L in Hz, for a second-order loop with an active lead-lag loop filter can be described as [2,8]

$$\begin{aligned} B_L &= \int_0^\infty |H(j2\pi f)|^2 df \\ &= \frac{\omega_n}{2}\left(\zeta + \frac{1}{4\zeta}\right) \end{aligned} \tag{40}$$

The natural frequency, ω_n, and the damping factor, ζ, determine the tracking performance of the NCDLL. Increasing the natural frequency will decrease the loop response time. The noise loop bandwidth, B_L, determines the amount of noise power that effects the tracking loop. Therefore, loop bandwidth, B_L, is chosen by the designer so as to minimize the effects of noise balanced with adequate response time. Smaller loop bandwidths reduce noise levels, while increasing response time resulting in more sluggish responses.

2.2.4 NCDLL Tracking Performance Characteristics. Mean-square tracking error or tracking jitter is of particular interest in analyzing the performance of the NCDLL. The power spectrum of the variance of the steady state tracking error, δ, is given by [2,9]

$$S_\delta(f) = |H(j2\pi f)|^2 S_{n\epsilon'}(f) \tag{41}$$

where $H(s)|_{s=j2\pi f}$ is the closed-loop transfer function defined in Equation 37 and $S_{n\epsilon'}(f) = S_{n\epsilon}(f)/K_d^2$ is the two-sided power spectrum of the Gaussian noise process at the input to the loop model. $S_\delta(f)$ is approximately flat over the loop bandwidth, B_L, so that the

variance of δ can be expressed as

$$\begin{aligned}\sigma_\delta^2 &= \int_{-\infty}^{\infty} S_\delta(f)df \\ &= \int_{-\infty}^{\infty} S_{n\epsilon'}(f)|H(j2\pi f)|^2 df \\ &= \frac{S_{n\epsilon}(0)}{K_d^2}\int_{-\infty}^{\infty}|H(j2\pi f)|^2 df \end{aligned} \tag{42}$$

where the integral on the right side of the equation is defined as the two-sided noise bandwidth B_L in Hz. For $\Delta \leq 1.0$, the variance, known as tracking jitter, then has the form [8,18]

$$\sigma_\delta^2 = \frac{B_L\Delta}{2S/N_0}\left[1+\frac{2}{(2-\Delta)S/(N_0B_N)}\right] \tag{43}$$

where B_N represents the single-sided bandwidth of the BPFs and LPFs. Equation 43 indicates that as Δ decreases, the tracking jitter variance decreases. This is caused by the cancelation of noise due to the overlap in the early and late channels for $\Delta < 1.0$.

2.2.5 Performance of the NCDLL With Multipath. The NCDLL performs well in tracking the direct path signal with no multipath. However, when multipath delays are present within a range of (0,1.5] chips, tracking errors may become significant. This section examines the effects of multipath on NCDLL performance. It is assumed that only one reflected signal is present. Neglecting the navigation message, as done previously, the received signal with one reflection can be expressed as

$$\begin{aligned} r_{mp}(t) &= \sqrt{2P}a_0c(t-\tau_\circ)cos(2\pi f_\circ t+\theta_0) \\ &\quad +\sqrt{2P}a_1c(t-\tau_\circ-\alpha T_c)cos(2\pi f_\circ t+\theta_1)+n(t). \end{aligned} \tag{44}$$

where the second term represents the reflected signal with delay αT_c. As in the previous section, the received signal enters into the 'early' and 'late' branches of the NCDLL upon which it is mixed with a locally generated replica of the spreading code represented by $c(t-\hat{\tau_\circ}\pm\Delta T_c/2)$ in which the + sign indicates the 'early' branch and − sign the 'late'

branch giving the resulting signals

$$\begin{aligned} x_e(t) &= \sqrt{2P}a_0c(t-\tau_o)c(t-\hat{\tau_o}+\Delta T_c/2)cos(2\pi f_o t+\phi) \\ &\quad +\sqrt{2P}a_1c(t-\tau_o-\alpha T_c)c(t-\hat{\tau_o}+\Delta T_c/2)cos(2\pi f_o t+\phi)+n_e(t) \end{aligned} \quad (45)$$

and,

$$\begin{aligned} x_l(t) &= \sqrt{2P}a_0c(t-\tau_o)c(t-\hat{\tau_o}-\Delta T_c/2)cos(2\pi f_o t+\phi) \\ &\quad +\sqrt{2P}a_1c(t-\tau_o-\alpha T_c)c(t-\hat{\tau_o}-\Delta T_c/2)cos(2\pi f_o t+\phi)+n_l(t). \end{aligned} \quad (46)$$

Once again, for high DS/SS processing gain systems such as GPS, code self noise can safely be neglected. Therefore, the signals after bandpass filtering becomes

$$\begin{aligned} y_e(t) &= \sqrt{2P}a_0R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]cos(2\pi f_o t+\theta_0) \\ &\quad +\sqrt{2P}a_1R_c\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right]cos(2\pi f_o t+\theta_1)+n_e(t) \end{aligned} \quad (47)$$

and,

$$\begin{aligned} y_l(t) &= \sqrt{2P}a_0R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]cos(2\pi f_o t+\theta_0) \\ &\quad +\sqrt{2P}a_1R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right]cos(2\pi f_o t+\theta_1)+n_l(t) \end{aligned} \quad (48)$$

where $\delta = (\tau-\hat{\tau_o})/T_c$. The squaring and LPF operation produces three signal components for each branch which is a result of squaring the direct path signal and the reflected signal and a cross term of the direct path and reflected signal. Neglecting the noise terms, the signals can be expressed as

$$\begin{aligned} y_e^2 &= Pa_0^2R_c^2\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right] \\ &\quad +2Pa_0a_1cos(\theta_0-\theta_1)R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]R_c\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right] \\ &\quad +a_1^2R_c^2\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right] \end{aligned} \quad (49)$$

and

$$
\begin{aligned}
y_l^2 &= Pa_0^2R_c^2\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right] \\
&\quad +2Pa_0a_1cos(\theta_0-\theta_1)R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right] \\
&\quad +a_1^2R_c^2\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right] \qquad (50)
\end{aligned}
$$

for the 'early' and 'late' branches respectively. Differencing the two branches forms the S curve, $\epsilon(t,\delta) = y_l^2 - y_e^2$ which can be expressed as

$$
\begin{aligned}
\epsilon(t,\delta) &= Pa_0^2\left\{R_c^2\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]-R_c^2\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]\right\} \\
&\quad +2Pa_0a_1cos(\theta_0-\theta_1)\left\{R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right]\right. \\
&\quad \left.-R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]R_c\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right]\right\} \\
&\quad +Pa_1^2\left\{R_c^2\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right]-R_c^2\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right]\right\}. \qquad (51)
\end{aligned}
$$

where $\theta_0-\theta_1 = 2\pi f_o\alpha T_c$. The first and third terms represent the direct-path and multipath components, respectively. The second term is the cross term component caused by the interaction of the direct-path and multipath signal. Figure 7 illustrates the influence of the multipath. It should be noted that the direct-path component is centered about $\delta = 0$. However, the net influence of the other components generated as a result of multipath, generates a tracking error, $\delta^* \neq 0$, corresponding to $\epsilon(\delta^*) = 0$.[2] The net result of the three components is illustrated in Figure 8. Here, one can see that the tracking point, corresponding to $\epsilon(\delta^*) = 0$, is no longer at $\delta = 0$, but has some multipath induced tracking error. The result is a steady-state code phase tracking error, δ^*, where $\delta^* = \frac{\tau_o-\hat{\tau_o}}{T_c}$. The amplitude of the code phase tracking error depends on the amplitude and phase parameters of the direct-path and multipath components: a_0, a_1, θ_0, and θ_1.[3]

The predicted steady-state tracking error can be determined by setting Equation 51 equal to zero and solving for δ for any $\alpha \in [0, 1.5]$. Figure 9 provides the predicted steady-

[2] δ^* represents the actual tracking point as opposed to the desired tracking point.
[3] Recall that $\alpha 2\pi f_o T_c = \theta_0 - \theta_1$.

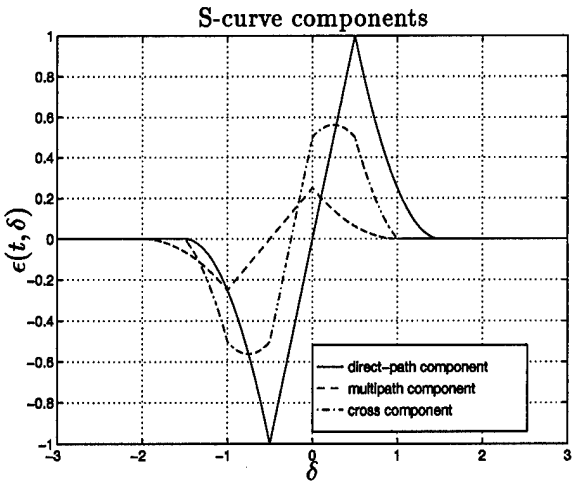


Figure 7 Typical NCDLL discriminator output components in the presence of multipath.

state tracking error[4] for $a_0 = 1.0$, $a_1 = 0.5$, $P = 0.5$, and $f_o T_c = 10$. As presented by van Nee (for $\Delta = 1.0$), multipath delays of up to 1.5 chips can cause errors in the tracking loop estimates [19]. The figure also illustrates that maximum tracking errors occur when a multipath signal is in-phase or out-of-phase with the line of sight signal, $\theta_0 - \theta_1 = \pi n$, where n is any integer. The estimate of the carrier phase, however, will theoretically have no tracking error, because carrier tracking follows the combination of the direct-path and multipath signal which have an equivalent zero crossing point when they are in-phase or 180° out of phase [15,19]. GPS has a product of $f_o T_c = 1540$. Therefore, the number of cycles of the tracking error curve would be much higher, within the same tracking error envelope; the envelope around the tracking error curve indicates the worst-case (i.e.,in-phase or out-of-phase multipath signal) for any product of $f_o T_c$.

2.2.6 Narrow Correlator Spacing. One of the early advances in receiver design to combat multipath was proposed by A.J. Van Dierendonck, Pat Fenton, and Tom Ford in the early 1990's [18]. They proposed narrowing the chip spacing in the NCDLL. This came about after evaluating the traditional design of the NCDLL.

[4]Note: The envelope around tracking error curve indicates worst-case values for any product of $f_o T_c$.

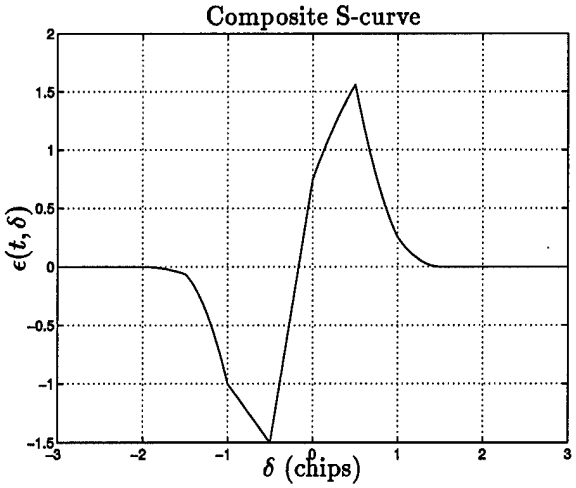


Figure 8 Typical NCDLL discriminator output in the presence of multipath

The improvements obtained by narrow correlator spacing of $0.1T_c$ require the use of a wide-bandwidth receiver design to sharpen the peak of the code correlation function. Due to the close spacing of the reference codes, the noise in the 'early' and 'late' correlator outputs is highly correlated. As such, the noise is reduced when the 'early' and 'late' paths are differenced. In addition, in the presence of multipath, the error tracking envelope is smaller which leads to improved code tracking performance [20].

In the traditional design, it was assumed that a one chip spacing provided optimum performance. In the previous designs, analog hardware was used in GPS receivers. A one chip spacing minimized hardware requirements. Also, early receivers were designed for receiving P-code which has a short chip width. Implementing narrower spacings makes the delay lock loop discriminator very narrow. It was feared that Doppler shift and other disturbances would cause a loss of code lock. Narrower spacing also requires faster clock rates to generate code replicas for the early and late gates. Previously, technology limited the ability to adjust for these considerations. Today's technology allows the use of narrower chip spacing between the early and late gate for C/A code [18]. As shown in Figure 10, the slope of the tracking curve is much steeper for $\Delta = 0.1$ than the slope for $\Delta = 1.0$ which indicates the requirement of a more dynamic VCC.

A closed form solution of the tracking envelope for the coherent DLL has been derived which can be used to examine the benefits of narrow correlator spacing. The solution

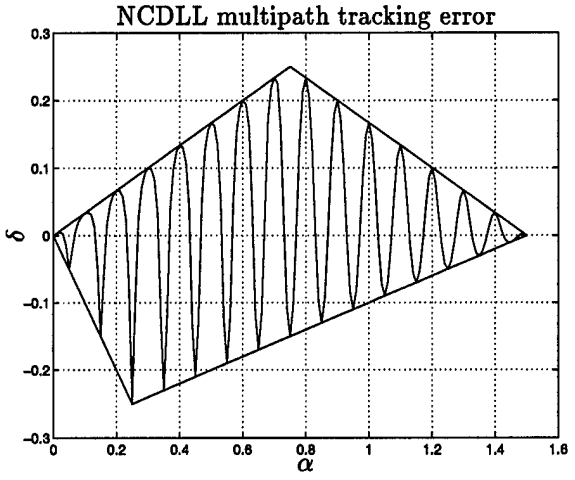


Figure 9 Predicted NCDLL multipath tracking error for parameters: $a_0 = 1.0$, $a_1 = 0.5$, $P = 0.5$, $f_o T_c = 10$, and $\Delta = 1.0$.

can be used to analyze the NCDLL as well since the worst-case tracking errors for a noncoherent DLL are the same as for a coherent DLL [15]. By analyzing the correlation function for the direct path signal in conjunction with the reflected signal, four sets of equations can be obtained, summarized in Table 1, through algebraic manipulation [15]. These equations are based on the interaction of the direct path correlation function with the reflected signals correlation function through four regions. The equations are valid for

Table 1 Equations for determining the code tracking phase error envelope of a coherent or NCDLL.

Range of α	δ
$0 \le \alpha < \frac{(a_0+a_m)\Delta}{2a_0}$ $(= a)$	$\frac{a_m \alpha}{a_0+a_m}$
$\frac{(a_0+a_m)\Delta}{2a_0} \le \alpha < T_c - \Delta\left(1 - \frac{(a_0+a_m)}{2a_0}\right)$ $(= b)$	$\frac{a_m \Delta}{2a_0}$
$T_c - \Delta\left(1 - \frac{(a_0+a_m)}{2a_0}\right) < \alpha \le T_c + \frac{\Delta}{2}$	$\frac{a_m}{2a_0 - a_m}\left(T_c + \frac{\Delta}{2} - \alpha\right)$
$\alpha > T_c + \frac{\Delta}{2}$	0

the condition that $\Delta \le T_c$ and $-a_0 < a_m \le a_0$. The constant a_m corresponds to $\pm a_1$ depending upon whether the reflected signal is in phase or out of phase with the direct path signal. Figure 11 displays the points of the tracking error envelope generated from Table 1 which, in conjunction with Figure 11, can be used to describe the tracking error

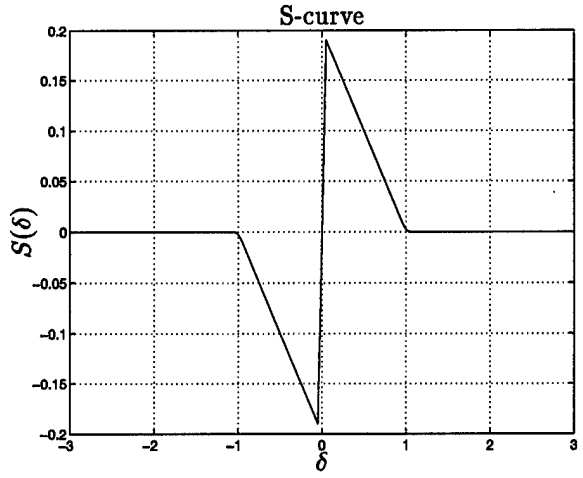


Figure 10 S-curve for NCDLL, $\Delta = 0.1$

envelope for any single multipath reflection with any value of Δ. Figure 11 illustrates the reduction in tracking error afforded by narrowing the correlator spacing, Δ.

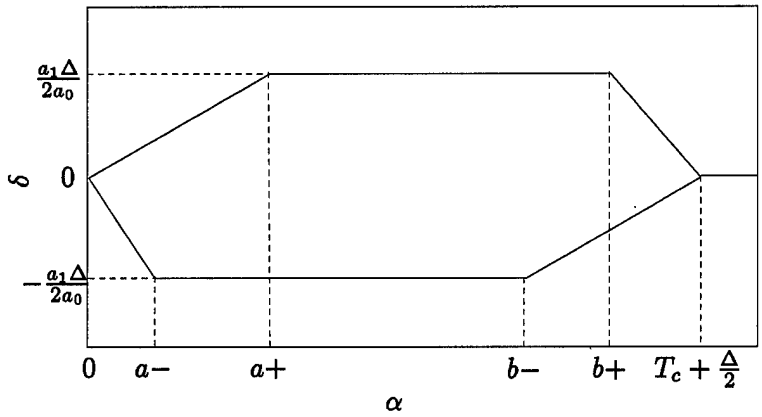


Figure 11 Code tracking error envelope (worst-case) in terms of multipath parameters and Δ. Note: $\pm a$ and $\pm b$ are the points relative to $\pm a_1$ for the equations in Table 1.

Figure 12 provides the results of narrowing Δ to 0.1 with the same signal parameters as used in Figure 9. Upon comparison, the worst case tracking error is reduced by an order of magnitude and the range of α which produces tracking error has been reduced to (0,1.05]. Note that the frequency of the tracking error curve is the same as in Figure 9, $f_o T_c = 10$. Figure 13 provides an overlay of the tracking error envelopes for $\Delta = 1.0$ and $\Delta = 0.1$ which clearly indicates the significant improvement afforded by narrowing

the correlator spacing. As such, the narrow correlator design of the NCDLL provides a

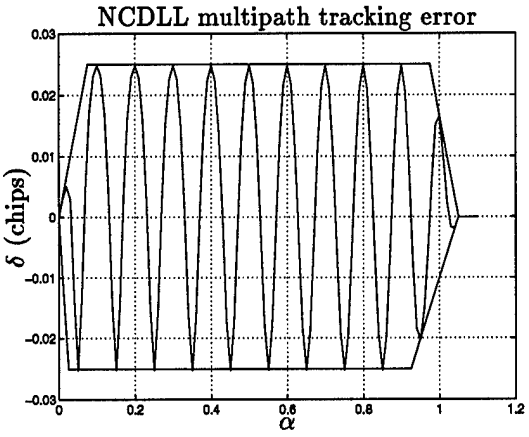


Figure 12 Predicted NCDLL multipath tracking error for parameters: $a_0 = 1.0$, $a_1 = 0.5$, $P = 0.5$, $f_o T_c = 10$, and $\Delta = 0.1$.

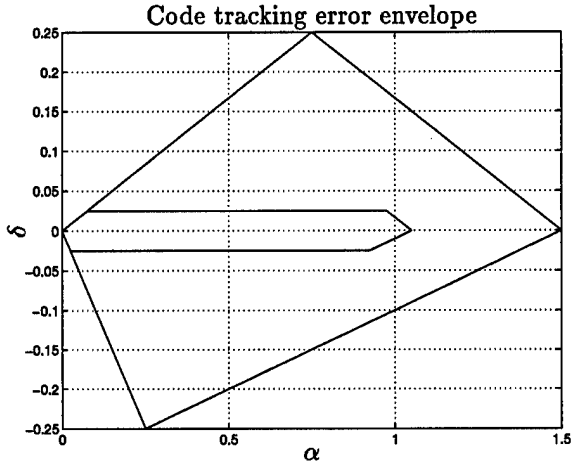


Figure 13 Code tracking error envelopes for $\Delta = 1.0$ and $\Delta = 0.1$

very good baseline for evaluation of other multipath mitigation techniques. These NCDLL properties have been documented by many sources including [8,9,12]. The NCDLL with narrow correlator spacing will be used as a performance baseline this thesis as well.

2.3 Estimator Designs

Although the NCDLL with narrow correlator spacing has had a profound effect on multipath reduction, better receiver designs have been achieved using estimators to determine the parameters of multipath signals in an effort to totally remove the error

caused by multipath. Three designs using estimation theory to remove the effects of multipath in a DS/SS or Code Division Multiple Access (CDMA) environment are the multipath estimating DLL (MEDLL), RAKE DLL (RDLL), and the Modified RAKE DLL (MRDLL).

2.3.1 MEDLL. The MEDLL was introduced by Richard van Nee to improve the performance of positioning solutions in a multipath environment [16, 17]. This design is based on Maximum Likelihood Estimation (MLE) theory which is used to estimate the phase, propagation delay, and amplitude of the direct path and reflected signals. One method of implementing MEDLL is accomplished by using a bank of M+1 correlators, where M is the number of reflected signals, to simultaneously estimate the parameters of the line-of-sight signal as well as the reflected signals [16]. The MEDLL determines a set of reference correlation functions with a certain amplitude, phase, and delay which gives the best estimate of the input correlation function. Each estimated multipath correlation function is subtracted from the measured correlation function which provides a remaining estimate of the direct path correlation function. A standard coherent early-late DLL proceeds this process to provide an estimate of the code loop tracking error. Essentially, all equations look the same as in the case of a conventional coherent spread spectrum receiver except for the estimation and removal of multipath signals [14, 16]. In normal operation the MEDLL is configured to estimated the parameters of the direct path signal and two reflected signals [14].

2.3.2 RAKE Delay Lock Loop. The basic configuration of the DLL was not intended to be used under adverse conditions such as envelope fading, code delay spread, and code Doppler spread. Stuber and Sheen [10, 13] proposed a new low complexity tracking loop for direct-sequence spread-spectrum signaling on multipath fading inter-symbol interference (ISI) channels found in applications such as code-division-multiple-access (CDMA) mobile radio communications. The new tracking loop, called the RDLL, exploits the inherent multipath diversity of the channel similar to a RAKE receiver. Although not intended for GPS, the RDLL has been shown to have excellent performance as compared with the traditional DLL for ranging applications such as mobile radio communications [10, 13].

The RDLL consists of a channel parameter estimation unit and a coherent tracking loop. The channel parameter estimation unit is composed of L+1 branches spaced at integer multiples of T_c which provide estimates to the correlator branches after mixing. The channel parameter estimation is a moving average filter that averages the received signal following despreading and demodulating. The best averaging length can be empirically determined and is based on the Doppler shift range of values of the channel [13]. The coherent tracking loop consists of L+1 correlator functions which use an 'early-late' code correlation process. The correlated signals are converted to baseband and summed to form a low pass error signal. This signal drives a Voltage Control Clock (VCC) which corrects the code phase error of the locally generated pseudo-noise code [10,13].

2.3.3 Modified RAKE Delay Lock Loop. The MRDLL, a design proposed first proposed by Laxton and DeVilbiss, uses maximum likelihood signal estimation to determine the signal parameters of the direct path signal and a single reflected signal [6,7]. The MRDLL, shown in Figure 14 is a modified version of the RDLL introduced by Sheen and Stuber. It is composed of three main components: the multiple-correlator tracking loop (MCTL), the adaptive loop controller (ALC), and the multiple or multipath correlator estimation unit (MCEU).

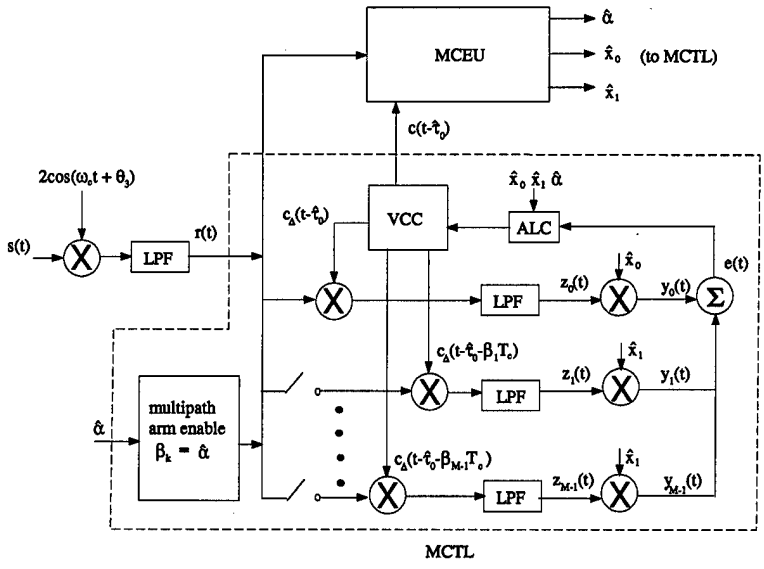


Figure 14 Modified RAKE delay lock loop (MRDLL) [6,7].

After conversion to baseband, the received direct path signal is tracked by the MCTL. The MCTL relies on the MCEU generated maximum likelihood estimated signal parameters to accurately track the direct path code phase in the presence of multipath. It was conceived as a bank of M tracking arms of which one is designated for the direct path signal. The remaining tracking arms are delayed by some range of values from the direct path tracking arm. In implementation a movable arm was actually used. The discriminator output is formed by summing the direct path arm with the arm that has been determined to contain the most accurate estimate of the correlation function of the reflected signal. The MCEU is composed of two components: a bank of M correlators and an estimator. It is designed to provide the MCTL with maximum likelihood estimates of the multipath delay with respect to the direct path delay as well as coefficients of the LOS baseband signal and the reflected baseband signal. These estimates allow the MCTL to remove the tracking error introduced by the reflected signal. Whereas the RDLL models the relative delays between the direct path and reflected signals as integer multiplies of the spreading code, it has been shown that smaller delays of fractions of a chip cause tracking errors in GPS applications. Therefore, the MCEU incorporates chip spacings in a range of (0, 1.5]. The MCEU feeds the estimate of the direct path signal coefficient, $\hat{x}_0$, to the direct path arm in the MCTL. It also feeds the estimate of the reflected signal coefficient, $\hat{x}_1$, to the movable tracking arm which is then used to provide the most accurate estimate of the correlation function of the reflected signal. Estimates of both signal coefficients as well as the estimated delay of the reflected signal are provided to the ALC.

Enhancements to the MRDLL have improved its performance. The enhanced MRDLL (eMRDLL) serves as the basic design for which performance of maximum likelihood estimation techniques can be analyzed. The methodology for the eMRDLL is presented in Chapter 3.

III. Enhanced Modified RAKE Delay Lock Loop (eMRDLL)

3.1 Overview

The enhanced modified RAKE delay lock loop (eMRDLL), shown in Figure 15, is composed of three main components: the multiple-correlator tracking loop (MCTL), the adaptive loop controller (ALC), and the multiple or multipath correlator estimation unit (MCEU). This coherent design requires the received signal to be converted to baseband prior to or during the correlation processes within the MCTL and the MCEU.

The 'early' 'late' gate tracking loops of the MCTL employ the same basic characteristics as a coherent DLL with the added capability of correcting the tracking process according to the estimated amplitudes and delays of the received LOS and multipath signals via the MCEU. The ALC dynamically adjusts the loop tracking response according to the estimated signal parameters.

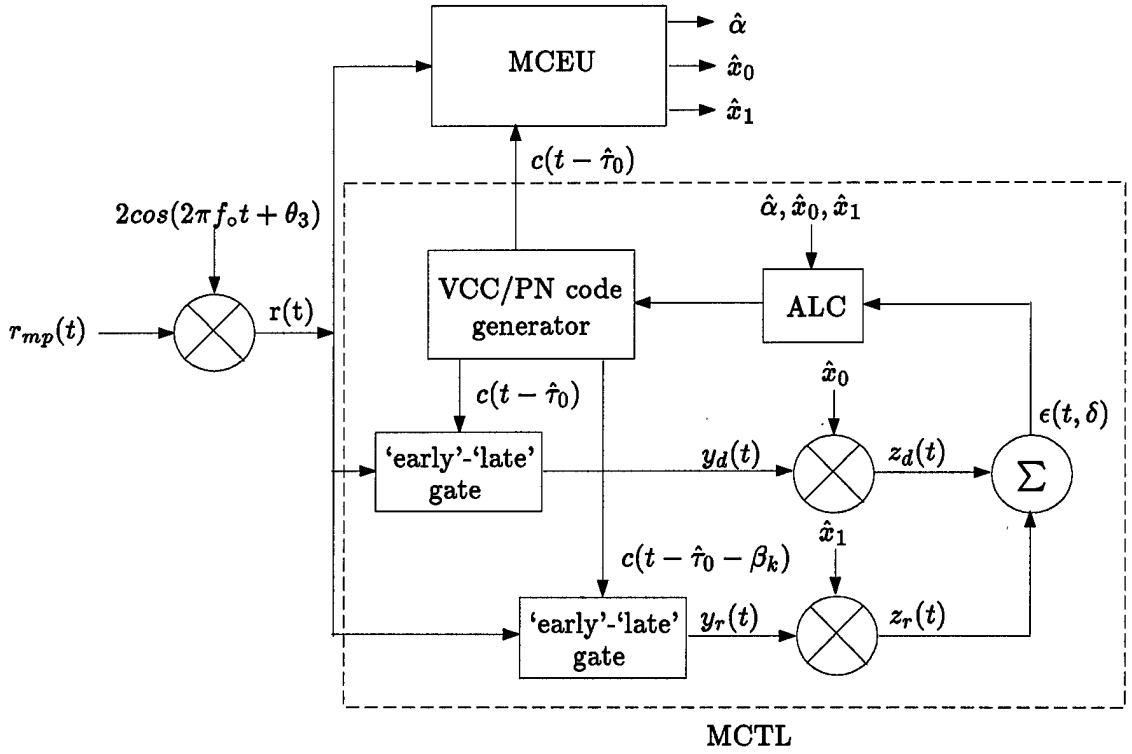


Figure 15 Modified RAKE delay lock loop (MRDLL) block diagram.

3.2 Baseband Conversion Process

The first processing stage of eMRDLL converts the received multipath signal to baseband. The received multipath signal, $r_{mp}(t)$, can be expressed as follows

$$\begin{aligned} r_{mp}(t) &= \sqrt{2P}a_0c(t-\tau_o)cos(2\pi f_o t+\theta_0) \\ &\quad +\sqrt{2P}a_1c(t-\tau_o-\alpha T_c)cos(2\pi f_o t+\theta_1)+n(t) \end{aligned} \tag{52}$$

where n(t) represents the noise expressed in Equation 3. As part of the baseband conversion process, the signal is mixed with a locally generated signal, $2cos(2\pi f_o+\theta_3)$, produced by a phase lock loop (PLL), and then low pass filtered in the MCTL and MCEU to convert the multipath signal to baseband. This represents a modification in the baseband conversion process as presented by Laxton and DeVilbiss, which contained a pre-correlation filter to convert the received signal to baseband prior to entering the MCTL and MCEU as shown in Figure 14 in Chapter 2 [6,7]. Using a pre-correlation filter is not necessary since the mixed signal will be low pass filtered within the MCTL and MCEU. In addition, low pass filtering prior to the correlation process may distort the signal, $r(t)$. Therefore, implementation of a pre-correlation filter would require the coefficients of the filter to be considered in the estimation process in order for the estimator in the MCEU to meet the Cramer Rao lower bound (CRLB) as a minimum variance unbiased (MVU) estimator [22].

The PLL signal can be generated by passing the input signal, $r_{mp}(t)$, through a correlator followed by a phase-locked loop (PLL) as shown in Figure 16 to produce the signal, $2cos(2\pi f_o t+\theta_3)$ [6].

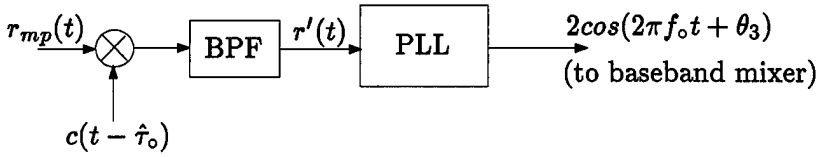


Figure 16 Typical carrier phase recovery scheme for GPS.

The signal, $r'(t)$, entering the PLL, obtained through the correlation process of mixing and bandpass filtering the received signal, can be expressed as

$$\begin{aligned} r'(t) &= \Big[\sqrt{2P}a_0 c(t-\tau_o)c(t-\hat{\tau}_o)cos(2\pi f_o t+\theta_0) \\ &\quad + \sqrt{2P}a_1 c(t-\tau_o-\alpha T_c)c(t-\hat{\tau}_o)cos(2\pi f_o t+\theta_1)\Big]_{BPF} \\ &= \sqrt{2P}a_0 R_c(0)cos(2\pi f_o t+\theta_0) + \sqrt{2P}a_1 R_c(\alpha)cos(2\pi f_o t+\theta_1) \\ &= \sqrt{2P}a_0 cos(2\pi f_o t+\theta_0) + \sqrt{2P}a_1 R_c(\alpha)cos(2\pi f_o t+\theta_1) \end{aligned} \tag{53}$$

Assuming perfect code phase synchronization, $\hat{\tau}_o = \tau_o$, an expression for θ_3 can be derived by the following technique [23]. By replacing $2\pi f_o t+\theta_1$ with $2\pi f_o t+\theta_0+\theta_1-\theta_0$ and using the trigonometric identity $cos(u \pm v) = cos(u)cos(v) \mp sin(u)sin(v)$, Equation 53 can be rewritten as

$$\begin{aligned} r'(t) &= \sqrt{2P}a_0 cos(2\pi f_o t+\theta_0) + \sqrt{2P}a_1 R_c(\alpha)cos(\theta_1-\theta_0)cos(2\pi f_o t+\theta_0) \\ &\quad -\sqrt{2P}a_1 R_c(\alpha)sin(\theta_1-\theta_0)sin(2\pi f_o t+\theta_0) \end{aligned} \tag{54}$$

Applying vector analysis, as illustrated in Figure 17, Equation 54 can be written as [6]

$$r'(t) = C(\alpha)cos(2\pi f_o t+\theta_0+\phi_e) \tag{55}$$

where $C(\alpha)$ is the magnitude of the combined signals and ϕ_e is the phase deviation due to the reflected signal and is given by

$$\begin{aligned} \phi_e &= tan^{-1}\left[\frac{\sqrt{2P}a_1 R_c(\alpha)sin(\theta_1-\theta_0)}{\sqrt{2P}a_0+\sqrt{2P}a_1 R_c(\alpha)cos(\theta_1-\theta_0)}\right] \\ &= tan^{-1}\left[\frac{-R_c(\alpha)sin(\theta_0-\theta_1)}{a_0/a_1+R_c(\alpha)cos(\theta_0-\theta_1)}\right] \end{aligned} \tag{56}$$

where $\theta_0-\theta_1 = 2\pi f_o \alpha T_c$. From Equation 55, the phase of the signal from the PLL, θ_3, can be expressed as $\theta_3 = \theta_0+\phi_e$.

Equation 56 provides insight into the effects of the the code phase and amplitude of the reflected signal. Figure 18 plots ϕ_e versus α for $a_0/a_1 = 2$ and $f_o T_c = 10$. The

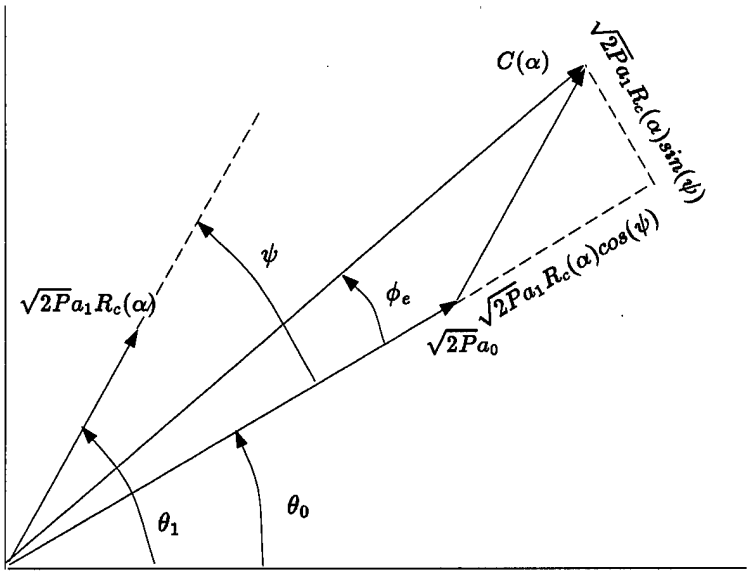


Figure 17 Vector representation of $r'(t)$. Note: $\psi = \theta_1 - \theta_0$.

phase error envelope shares properties with the triangular code auto correlation function, R_c, of Equation 8. Also, the carrier phase of the PLL generated signal is in-phase with the direct-path carrier (i.e., $\phi_e = 0$) whenever α is at delays that corresponds to $2\pi f_c \alpha T_c = \pi n$ where n is an integer. This corresponds with the reflected signal being in-phase or 180 degrees out of phase with the direct-path signal.

3.3 MCTL Operation

Mixing the received signal, $r_{mp}(t)$, with the signal from the PLL, $2cos(2\pi f_o t + \theta_3)$, produces

$$\begin{aligned} r(t) &= \sqrt{2P}a_0 c(t-\tau_o)cos(\theta_0 - \theta_3) + \sqrt{2P}a_0 c(t-\tau_o)cos(4\pi f_o t + \theta_0 + \theta_3) \\ &\quad + \sqrt{2P}a_1 c(t-\tau_o - \alpha T_c)cos(\theta_1 - \theta_3) + \sqrt{2P}a_1 c(t-\tau_o - \alpha T_c)cos(4\pi f_o t + \theta_1 + \theta_3) \\ &\quad + 2n(t)cos(2\pi f_o t + \theta_3) \end{aligned} \tag{57}$$

which enters the MCEU and the MCTL. The 'early-late' gates of the MCTL operate similar to the 'early' and 'late' branches of a coherent DLL, in which the received signal is mixed and filtered to baseband. eMRDLL is unique in that it implements a floating 'early-late'

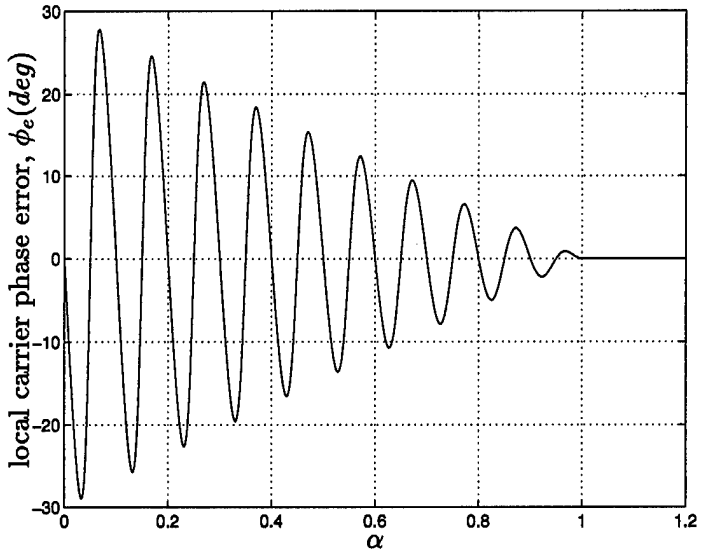


Figure 18 Local carrier phase error, ϕ_e, for $a_0/a_1 = 2$ and $f_o T_c = 10$.

gate code tracking loop for the reflected signal in addition to the 'early-late' gate tracking loop for the direct path signal. The floating 'early-late' code tracking loop uses estimates of the multipath parameters provided by the MCEU to synchronize the correlators of the 'early' and 'late' branches. Unlike the traditional DLL, the tracking loops of the MCTL are able to synchronize to the reflected signal as well as the direct path signal to reduce code phase tracking error.

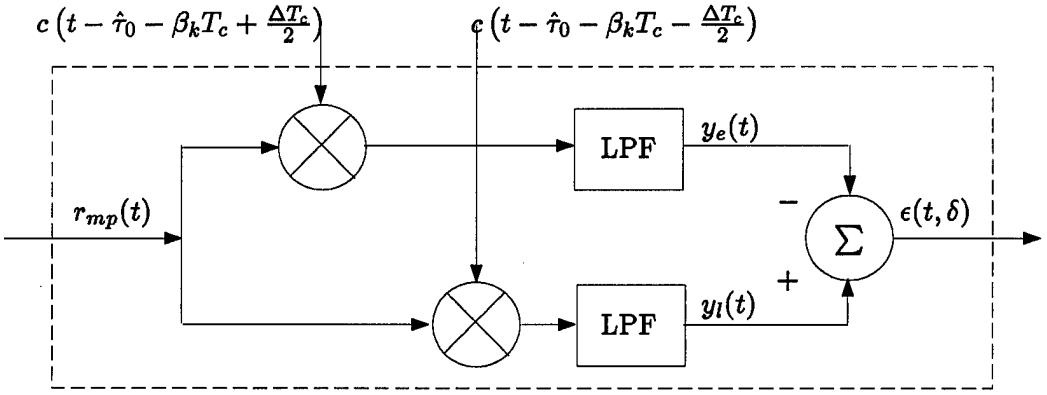


Figure 19 'Early' 'late' gate delay lock loop block diagram.

Since the signal, $r(t)$, is low pass filtered in the MCTL as well as the MCEU, it can be simplified as

$$r(t) = x_0 c(t - \tau_o) + x_1 c(t - \tau_o - \alpha T_c) + n'(t) \tag{58}$$

where

$$\begin{aligned} x_0 &= \sqrt{2P} a_0 cos(\theta_0 - \theta_3) \\ x_1 &= \sqrt{2P} a_1 cos(\theta_1 - \theta_3) \end{aligned} \tag{59}$$

and $n'(t)$ is lowpass, zero-mean, AWGN given as

$$n'(t) = \sqrt{2}[n_I(t) cos(\theta_3) + n_Q(t) sin(\theta_3)]. \tag{60}$$

Upon entering into the 'early' and 'late' branches of the tracking loop, the signal, $r(t)$, is mixed with the locally generated spreading code defined as

$$c_\Delta(t - \hat{\tau}_o - \beta_k T_c) \triangleq c(t - \hat{\tau}_o - \beta_k T_c - \frac{\Delta T_c}{2}) - c(t - \hat{\tau}_o - \beta_k T_c + \frac{\Delta T_c}{2}) \tag{61}$$

where MRDLL has traditionally implemented a one chip spacing, $\Delta = 1.0$. After mixing, the signals in each branch are lowpass filtered.

3.3.1 Direct Path Channel. For high processing gains, such as GPS, and small loop bandwidths ($B_L \leq 10\ Hz$), the filtered signals of the 'early' and 'late' branch of the direct path channel, for which $\beta_k T_c = 0$, can be expressed as [9,11]

$$\begin{aligned} y_{d,e}(t) &= \Big[x_0 c(t - \tau_o) c(t - \hat{\tau}_o + \frac{\Delta}{2} T_c) + x_1 c(t - \tau_o - \alpha T_c) c(t - \hat{\tau}_o + \frac{\Delta}{2} T_c) \\ &\quad + n'(t) c(t - \hat{\tau}_o + \frac{\Delta}{2} T_c) \Big]_{LPF} \\ &= x_0 R_c \left[\left(\delta + \frac{\Delta}{2} \right) T_c \right] + x_1 R_c \left[\left(\delta + \alpha + \frac{\Delta}{2} \right) T_c \right] \\ &\quad + \left[n'(t) c(t - \hat{\tau}_o + \frac{\Delta}{2} T_c) \right]_{LPF} \end{aligned} \tag{62}$$

and

$$
\begin{aligned}
y_{d,l}(t) &= \Big[x_0 c(t-\tau_\circ)c(t-\hat{\tau}_\circ-\frac{\Delta}{2}T_c)+x_1 c(t-\tau_\circ-\alpha T_c)c(t-\hat{\tau}_\circ-\frac{\Delta}{2}T_c) \\
&\quad +n'(t)c(t-\hat{\tau}_\circ-\frac{\Delta}{2}T_c)\Big]_{LPF} \\
&= x_0 R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]+x_1 R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right] \\
&\quad +\left[n'(t)c(t-\hat{\tau}_\circ-\frac{\Delta}{2}T_c)\right]_{LPF} \qquad (63)
\end{aligned}
$$

respectively. Differencing the 'early' and 'late' branches, $y_{d,l}-y_{d,e}$, produces

$$
\begin{aligned}
y_d(t) &= x_0\left\{R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]\right\} \\
&\quad +x_1\left\{R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right]\right\} \\
&\quad +\left[n'(t)c_\Delta(t-\hat{\tau}_\circ)\right]_{LPF} \qquad (64)
\end{aligned}
$$

After the 'early-late' differencing operation, the signal enters the *gain-phase correlators* upon which the signal is mixed with the MCEU signal parameter estimate of $\hat{x}_0$; the output of this direct-path arm can be expressed as

$$
\begin{aligned}
z_d(t,\delta) &= x_0\hat{x}_0\left\{R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]\right\} \\
&\quad +\hat{x}_0 x_1\left\{R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\alpha+\frac{\Delta}{2}\right)T_c\right]\right\} \\
&\quad +\hat{x}_0\left[n'(t)c_\Delta(t-\hat{\tau}_\circ)\right]_{LPF}. \qquad (65)
\end{aligned}
$$

3.3.2 Multipath Channel. Similarly, the mixed signals for the 'early' and 'late' branch of the reflected path channel can be expressed as follows

$$\begin{aligned} y_{r,e}(t) &= \Big[x_0 c(t-\tau_o)c(t-\hat{\tau}_o-\beta_k T_c+\frac{\Delta}{2}T_c) \\ &\quad + x_1 c(t-\tau_o-\alpha T_c)c(t-\hat{\tau}_o-\beta_k T_c+\frac{\Delta}{2}T_c) \\ &\quad + n'(t)c(t-\hat{\tau}_o-\beta_k T_c+\frac{\Delta}{2}T_c)\Big]_{LPF} \\ &= x_0 R_c\left[\left(\delta-\alpha+\frac{\Delta}{2}\right)T_c\right]+x_1 R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right] \\ &\quad + \left[n'(t)c(t-\hat{\tau}_o-\beta_k T_c+\frac{\Delta}{2}T_c)\right]_{LPF} \end{aligned} \tag{66}$$

and

$$\begin{aligned} y_{r,l}(t) &= \Big[x_0 c(t-\tau_o)c(t-\hat{\tau}_o-\beta_k T_c-\frac{\Delta}{2}T_c) \\ &\quad + x_1 c(t-\tau_o-\alpha T_c)c(t-\hat{\tau}_o-\beta_k T_c-\frac{\Delta}{2}T_c) \\ &\quad + n'(t)c(t-\hat{\tau}_o-\beta_k T_c-\frac{\Delta}{2}T_c)\Big]_{LPF} \\ &= x_0 R_c\left[\left(\delta-\alpha-\frac{\Delta}{2}\right)T_c\right]+x_1 R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right] \\ &\quad + \left[n'(t)c(t-\hat{\tau}_o-\beta_k T_c-\frac{\Delta}{2}T_c)\right]_{LPF} \end{aligned} \tag{67}$$

respectively. Differencing the 'early' and 'late' branches, $y_{r,l}-y_{r,e}$, produces

$$\begin{aligned} y_r &= x_0\left\{R_c\left[\left(\delta-\alpha-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta-\alpha+\frac{\Delta}{2}\right)T_c\right]\right\} \\ &\quad + x_1\left\{R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]\right\} \\ &\quad + \left[n'(t)c_\Delta(t-\hat{\tau}_o-\beta_k T_c)\right]_{LPF}. \end{aligned} \tag{68}$$

After the 'early-late' differencing operation, the signal enters the *gain-phase correlators* upon which the signal is mixed with the MCEU signal parameter estimate $\hat{x}_1$. The output

of the multipath arm can be expressed as

$$\begin{aligned} z_r(t,\delta) &= x_0\hat{x}_1\left\{R_c\left[\left(\delta+\alpha-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta-\alpha+\frac{\Delta}{2}\right)T_c\right]\right\} \\ &\quad +x_1\hat{x}_1\left\{R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]\right\} \\ &\quad +\hat{x}_1\left[n'(t)c_\Delta(t-\hat{\tau}_o-\beta_kT_c)\right]_{LPF} \end{aligned} \tag{69}$$

3.3.3 Discriminator Output. Under the assumption that the MCEU provides perfect estimates of $\hat{x}_0$ and $\hat{x}_1$ to the *gain-phase correlators*, the respective outputs of the direct-path and multipath branches are [6]

$$\begin{aligned} z_d(t,\delta) &= x_0^2D(\delta)+x_0x_1D(\delta+\alpha)+x_0\left[n'(t)c_\Delta(t-\hat{\tau}_o)\right]_{LPF} \\ z_r(t,\delta) &= x_0x_1D(\delta-\alpha)+x_1^2D(\delta)+x_1\left[n'(t)c_\Delta(t-\hat{\tau}_o-\beta_kT_c)\right]_{LPF} \end{aligned} \tag{70}$$

where the MCTL $D-curve$ is defined as

$$D(\delta)\triangleq R_c\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right]-R_c\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right] \tag{71}$$

Summing the outputs of the direct-path channel and the multipath channel produces the code tracking discriminator/error signal

$$\begin{aligned} e(t) &= (x_0^2+x_1^2)D(\delta)+x_0x_1[D(\delta+\alpha)+D(\delta-\alpha)] \\ &\quad +x_0\left[n'(t)c_\Delta(t-\hat{\tau}_o)\right]_{LPF}+x_1\left[n'(t)c_\Delta(t-\hat{\tau}_o-\beta_kT_c)\right]_{LPF} \end{aligned} \tag{72}$$

which can be expressed as [6]

$$e(t,\delta)=S(\delta)+n_e(t) \tag{73}$$

where $S(\delta)$ represents the MRDLL S-curve given as

$$S(\delta)=(x_0^2+x_1^2)D(\delta)+x_0x_1[D(\delta+\alpha)+D(\delta-\alpha)] \tag{74}$$

and the discriminator output noise, $n_e(t)$, is given as

$$n_e(t) = x_0 \left[n'(t)c_\Delta(t - \hat{\tau}_o)\right]_{LPF} + x_1 \left[n'(t)c_\Delta(t - \hat{\tau}_o - \beta_k T_c)\right]_{LPF} \tag{75}$$

Figure 20 provides a plot of the discriminator components of the direct-path channel and the multipath channel. The figure also shows that the sum of the two channels produces the discriminator output which has an error signal equal to zero, under the condition of perfect estimation.

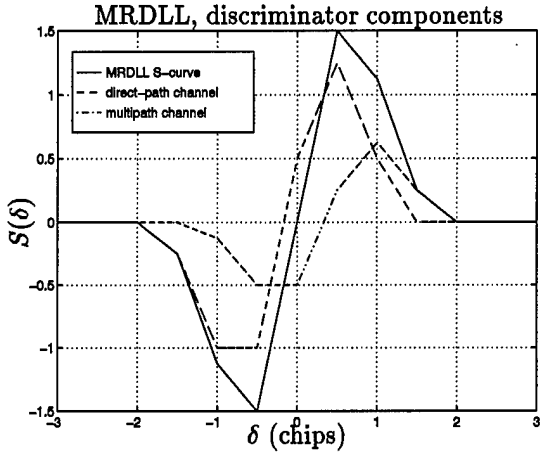


Figure 20 Typical MRDLL S-curve discriminator.

3.3.4 Determination of Noise, $n_e(t)$. The noise components of $n_e(t)$ were obtained under the assumption that the LPFs in each of the 'early-late' gates were ideal. Alternatively, the noise component, $n_e(t)$, can be represented as

$$n_e(t) = x_0 n_d(t) + x_1 n_r(t) \tag{76}$$

where $n_d(t)$ and $n_r(t)$ are described by [6]

$$\begin{aligned} n_k(t) &= \sqrt{2}\left[n_k^I(t)cos(\theta_3) + n_k^Q(t)sin(\theta_3)\right] \\ n_k^I(t) &\triangleq n_I(t)c_\Delta(t - \hat{\tau}_o - \beta_k T_c) * h_{LPF}(t) \\ n_k^Q(t) &\triangleq n_Q(t)c_\Delta(t - \hat{\tau}_o - \beta_k T_c) * h_{LPF}(t) \end{aligned} \tag{77}$$

where $k = d$ or r, $*$ denotes convolution, and $h_{LPF}(t)$ is the impulse response of the LPFs. For large code period, $N \gg 1$, and small loop bandwidth, $B_L < 10\ Hz$, the amplitude of the two-sided PSD, $S_{n'k}$, of $n_k^I(t)$ and $n_k^Q(t)$ is approximately N_0 in units W/Hz [9]. As such, the output of the noise of the k^{th} branch has a two-sided PSD

$$S_{n_k}(f) = \begin{cases} 2N_0 & |f| \leq LPF \\ 0 & \text{elsewhere} \end{cases} . \tag{78}$$

Under the condition that the MCEU provides perfect estimates to the *gain-phase correlators*, the combined PSD noise term of $z_d(t)$ and $z_r(t)$ is

$$S_{n_e}(f) = \begin{cases} 2N_0(x_0^2 + x_1^2) & |f| \leq LPF \\ 0 & \text{elsewhere} \end{cases} . \tag{79}$$

3.3.5 eMRDLL Linear Model and Tracking Performance. The operation of the VCC within the MCTL provides the same performance characteristics as described in Chapter 2 for the NCDLL. Therefore, the output phase of the VCC can be expressed as

$$\begin{aligned} \frac{\hat{\tau}_o(t)}{T_c} &= K_o \int_0^t e(\alpha, \delta) * f(\lambda) d\lambda \\ &= K_o \int_0^t \int_{-\infty}^{\alpha} e(\alpha, \delta) f(\alpha - \lambda) d\lambda d\alpha. \end{aligned} \tag{80}$$

where the output of the loop filter is described by its impulse response, $f(t)$, is convolved with the input signal, $e(t, \delta)$. Substituting Equation 73 into Equation 80 provides the equation for the nonlinear model

$$\frac{\hat{\tau}_o(t)}{T_c} = K_o \int_0^t \int_{-\infty}^{\alpha} (S(\delta) + n_e(\lambda))\, f(\alpha - \lambda) d\lambda d\alpha \tag{81}$$

where $\delta = (\tau_o(t) - \hat{\tau}_o(t))/T_c$.

For small tracking errors, the linear region of the S-curve can be expressed as [6]

$$S(\delta) = A\delta = A\left(\frac{\tau_o(t) - \hat{\tau}_o(t)}{T_c}\right) \tag{82}$$

where A is defined as the slope of the linear operating region for small δ. Substituting Equation 82 into the nonlinear equation, Equation 81, results in

$$\frac{\hat{\tau}_o(t)}{T_c} = AK_o \int_0^t \int_{-\infty}^{\lambda} \left[\left(\frac{\tau_o(\alpha) - \hat{\tau}_o(\alpha)}{T_c} \right) + \frac{n_e(\alpha)}{A} \right] f(\lambda - \alpha) d\alpha d\lambda \tag{83}$$

which, in the Laplace domain, can be represented as

$$\frac{\hat{\tau}_o(s)}{T_c} = \frac{\tau_o(s) - \hat{\tau}_o(s)}{T_c} \left[AK_o \frac{F(s)}{s} \right] \tag{84}$$

where $F(s)$ is the Laplace transform of f(t). Therefore, the linear equivalent circuit model for the MCTL can be represented by replacing the terms K_d and $n_e(t)$ from the model in Figure 6 with A and $n_e(t)$ as shown in Figure 21.

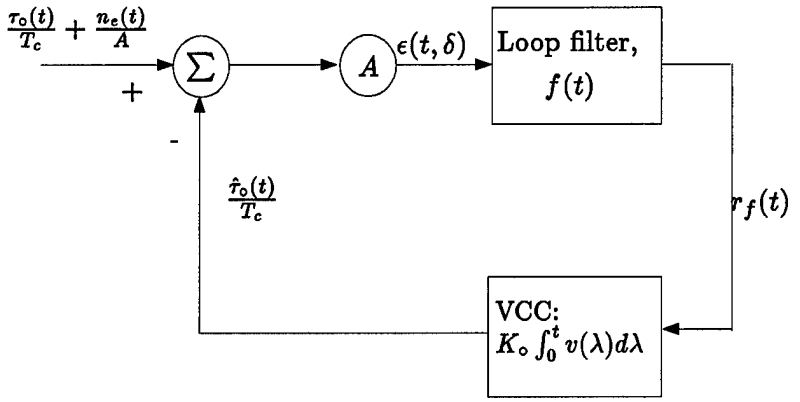


Figure 21 eMRDLL linear equivalent circuit.

Solving for $\hat{\tau}_o(s)/\tau_o(s)$ given Equation 84 results in the closed loop transfer function

$$H(s) \triangleq \frac{\hat{\tau}_o(s)}{\tau_o(s)} = \frac{AK_oF(s)}{s + AK_oF(s)} \tag{85}$$

which characterizes the eMRDLL for fixed estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$. Choosing an active lead-lag loop filter, the transfer function of Equation 85 can be transformed into the classical

model representation given as [2,9]

$$H(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{86}$$

where the loop natural frequency, ω_n in rad/sec, and the (unitless) damping factor, ζ, are defined as

$$\omega_n = \sqrt{\frac{AK_\circ}{\tau_1}} \tag{87}$$

and

$$\zeta = \frac{\tau_2}{2}\omega_n \tag{88}$$

respectively. As with the NCDLL, the single-sided noise equivalent bandwidth, B_L in Hz, for a second-order loop with an active lead-lag loop filter can be described as [2,8]

$$\begin{aligned} B_L &= \int_0^\infty |H(j2\pi f)|^2 df \\ &= \frac{\omega_n}{2}\left(\zeta + \frac{1}{4\zeta}\right). \end{aligned} \tag{89}$$

The power spectrum of the tracking jitter can be expressed as [2,9]

$$S_\delta(f) = |H(j2\pi f)|^2 S_{ne'}(f) \tag{90}$$

where $H(s)$ is the closed-loop transfer function defined in Equation 86, and $S_{ne'}(f) = S_{ne}(f)/A^2$ is the two-sided power spectrum of the Gaussian noise process at the input to the loop model. $S_\delta(f)$ is approximately flat over the loop bandwidth, B_L, so that the

variance of δ can be expressed as

$$\begin{aligned}
\sigma_\delta^2 &= \int_{-\infty}^{\infty} S_\delta(f)df \\
&= \int_{-\infty}^{\infty} S_{ne'}(f)|H(j2\pi f)|^2 df \\
&= \frac{S_{ne}(0)}{A^2}\int_{-\infty}^{\infty} |H(j2\pi f)|^2 df \\
&= \frac{4\left(x_0^2+x_1^2\right)N_0}{A^2}B_L
\end{aligned} \tag{91}$$

where the integral on the right side of the equation is defined as the two-sided noise bandwidth B_L in Hz.

As stated in Chapter 2, the natural frequency, ω_n, and the damping factor, ζ, determine the tracking performance of the code tracking loop and the noise loop bandwidth, B_L, determines the amount of noise power that affects the tracking loop. As shown in Equations 87 and 88, the slope of the linear region of the S-curve, A, which varies for different values of amplitude and delay of the received multipath signal, also determines the response characteristics of the loop via ω_n and ζ. As such, the changing value of A, as tabulated in Table 2, will also impact tracking performance.

3.4 ALC Operation

Because the transient response of the loop is dependent upon the parameters of the reflected signal, a variable gain is introduced in an attempt to eliminate the dependency. This is accomplished by preceding the loop filter with a variable gain, as done in the work of Laxton and DeVilbiss [6,7].

As shown in Figure 22, the ALC is a second order loop controller preceded by a variable gain. The variable gain allows the designer to fix the dynamic response of the MCTL within a linear operating region. The ALC is a unique element not present in the RDLL design of [10]. The ALC uses the parameters provided by the MCEU to control tracking performance relative to the discriminator output [6,7].

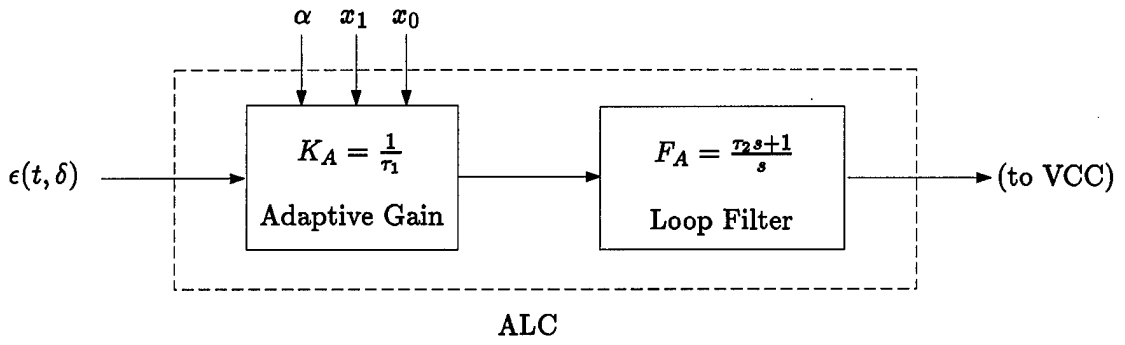


Figure 22 Adaptive loop controller (ALC) block diagram.

Assuming a 2^{nd} order active lead-lag loop filter, given as

$$F_A(s) = \frac{1+\tau_2 s}{s}, \tag{92}$$

the filter shall be preceded by a variable gain, $K_A = 1/\tau_1$. This provides a composite loop transfer function of

$$\begin{aligned} F(s) &= K_A \frac{\tau_2 s+1}{s} \\ &= \frac{\tau_2 s+1}{\tau_1 s}. \end{aligned} \tag{93}$$

As shown in Table 2 and illustrated in Figures 23 and 24, the slope of the linear operating region, A, varies with the signal parameters of $r_{mp}(t)$. The variable gain of the ALC allows adaptation to changing signal parameters so as to maintain a fixed natural frequency, ω_n, and, therefore, a fixed damping ratio and noise equivalent loop bandwidth, B_L.

The ALC uses received signal estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ from the MCEU to determine an estimate of the slope of the linear tracking region, $\hat{A}$, based on the equations provided in Table 2 [6,7]. The estimate, $\hat{A}$, is then used to adjust the gain

$$K_A = \frac{1}{\tau_1} = \frac{\omega_n^2}{\hat{A}K_o} \tag{94}$$

where ω_n and K_o are fixed parameters. The coefficient, τ_2, in the loop filter remains fixed based on Equation 88 which is specified by the designer based on desired performance characteristics.

Table 2 MCTL linear operating region.

Gain = (A/2)	Multipath Delay Range	Tracking Error Range
$x_0^2 + x_1^2 + 2x_0x_1$	$0 \leq 0.5$	$-0.5 + \alpha \leq \delta \leq 0.5 - \alpha$
$x_0^2 + x_1^2 + \frac{1}{2}x_0x_1$	$\alpha = 0.5$	$-0.5 \leq \delta \leq 0.5$
$x_0^2 + x_1^2 - x_0x_1$	$0.5 < 1.0$	$0.5 - \alpha \leq \delta \leq -0.5 + \alpha$
$x_0^2 + x_1^2 - x_0x_1$	$1.0 \leq 1.5$	$-1.5 + \alpha \leq \delta \leq 1.5 - \alpha$
$x_0^2 + x_1^2 - \frac{1}{2}x_0x_1$	$\alpha = 1.5$	$-0.5 \leq \delta \leq 0.5$

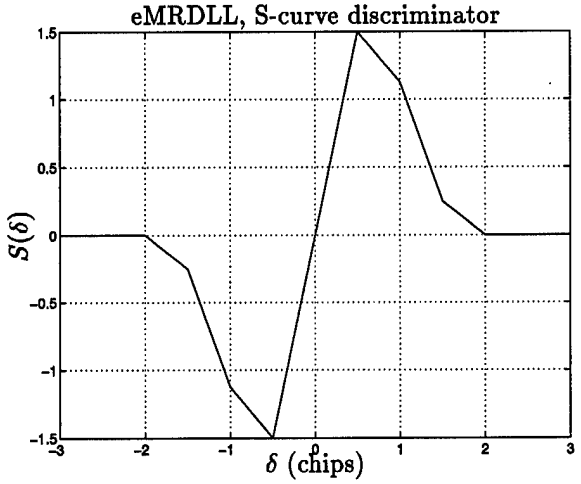


Figure 23 eMRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 0.5$.

3.5 MCEU Operation

The MCEU consists of a bank of M correlators followed by a ML estimation block as shown in Figure 25. Each correlator consists of a mixer followed by a low pass filter, which acts as an integrator when high processing gains are used. The estimation block generates estimates of the reflected signal delay relative to the direct path signal, as well as coefficients of the LOS and reflected baseband signals; these estimates are fed to the MCTL and the ALC. The MCEU is capable of estimating the parameters of multiple reflected signals. As such, the received signal, $r(t)$, entering the MCEU is represented as

$$r(t) = x_0 c(t - \tau_0) + \sum_{i=1}^{N} x_i c(t - \tau_0 - \alpha_i T_c) + n(t) \tag{95}$$

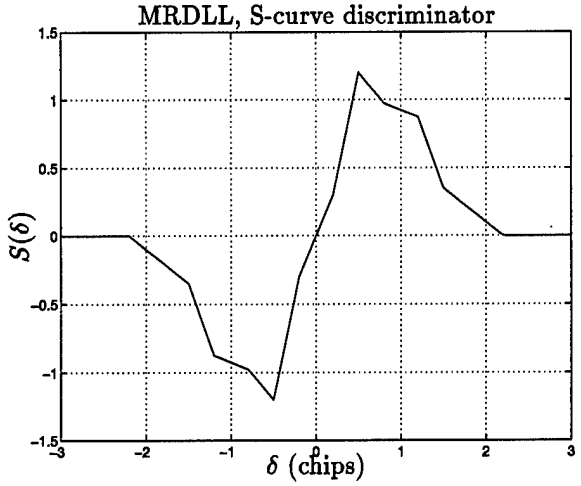


Figure 24 MRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 0.7$.

where $x_i = \sqrt{2P}a_i cos(\theta_i - \theta_P)$, and N represents the number of reflected signals.[1]

The received signal enters the bank of correlators, where each arm is uniformly spaced, at β_k, through the range of $[0, 1.5]$. Under perfect tracking conditions, where $\hat{\tau}_o = \tau_o$, the sampled output of the k^{th} correlator can be expressed as

$$R_k(nT_s) = x_0 R_c(\beta_k) + \sum_{i=1}^{N} x_i R_c(\alpha_i - \beta_k) \tag{96}$$

where β_k is the delay of the respective correlator arm, relative to $\hat{\tau}_o$, and T_s represents the sampling period of the estimator.

Let α be defined as

$$\alpha \equiv [\alpha_1 \; \alpha_2 \cdots \alpha_N]^T . \tag{97}$$

The sampled outputs from the bank of M correlators entering the estimator block are collectively

$$\mathbf{R} = [R_0 \; R_1 \cdots R_{M-1}]^T \tag{98}$$

[1] θ_P represents the phase of the signal generated from the PLL, which was previously represented as θ_3. It has been changed for this section to remove ambiguity with the number of multipath signals.

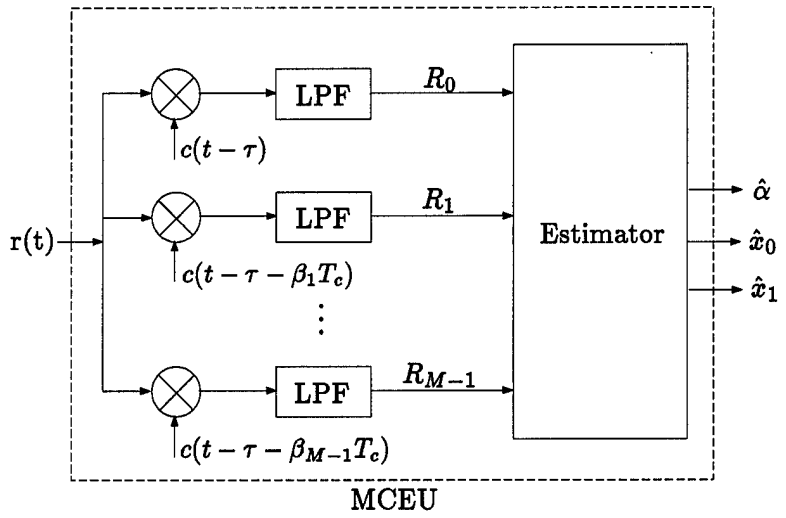


Figure 25 Multiple-correlator estimation unit (MCEU) block diagram.

which can be expressed as the linear statistical data model, $\mathbf{R} = \mathbf{H}(\alpha)\mathbf{x} + \mathbf{v}$. $\mathbf{H}(\alpha)$ is an $M \times N+1$ observation/regressor matrix where M is the number of correlator arms and $N+1$ is the total number of signal paths. $\mathbf{H}(\alpha)$ can be represented as

$$\mathbf{H}(\alpha) = \begin{bmatrix} R_c(\beta_0) & R_c(\alpha_1 - \beta_0) & \cdots & R_c(\alpha_N - \beta_0) \\ \vdots & \vdots & \ddots & \vdots \\ R_c(\beta_{M-1}) & R_c(\alpha_1 - \beta_{M-1}) & \cdots & R_c(\alpha_N - \beta_{M-1}) \end{bmatrix} \tag{99}$$

and the composite signals of the correlator branches are collectively

$$\mathbf{x} = [x_0 \cdots x_N]^T. \tag{100}$$

The observation noise vector, $\mathbf{v}$, is modeled [5] as zero mean Gaussian with covariance $\mathbf{C_v}$. To determine an expression for $\mathbf{C_v}$, the spreading code is considered a random binary process[2], independent of the observation noise. The continuous-time cross correlation between the i^{th} and j^{th} correlator noise outputs is

$$R_{ij}(\tau) = E[v_i(t)v_j(t+\tau)] \tag{101}$$

[2]This is a reasonable assumption, because a maximum length spreading code sequence is pseudorandom.

which, for narrowband low pass filtered noise, can be approximated as [6]

$$R_{ij}(\tau) \approx R_c(\tau + \Delta\beta_{ij})\sigma_v^2 \tag{102}$$

where $\Delta\beta_{ij} = \beta_i - \beta_j$ and σ_v^2 is the variance of each of the correlator noise outputs. As such, the instantaneous observation noise covariance matrix, $\mathbf{C_v} \equiv E\left[\mathbf{v(t)v^T(t)}\right] - E\left[\mathbf{v(t)}\right]E\left[\mathbf{v^T(t)}\right]$, can be expressed as

$$\mathbf{C_v} = \sigma_v^2\mathbf{C} \tag{103}$$

where $\mathbf{C}$ is the $M \times M$ symmetric, Toeplitz correlation matrix [6]

$$\mathbf{C} = \begin{bmatrix} 1 & R_c(\beta_0 - \beta_1) & \cdots & R_c(\beta_0 - \beta_{M-1}) \\ R_c(\beta_1 - \beta_0) & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & R_c(\beta_{M-2} - \beta_{M-1}) \\ R_c(\beta_{M-1} - \beta_0) & \cdots & R_c(\beta_{M-1} - \beta_{M-2}) & 1 \end{bmatrix}. \tag{104}$$

For a general linear data model, $\mathbf{R} = \mathbf{H}(\alpha)\mathbf{x} + \mathbf{v}$, where $\mathbf{H}(\alpha)$ is an $N \times p$ matrix of known structure and $\mathbf{v} \sim N(\mathbf{0}, \mathbf{C_v})$ is a noise vector of dimension $N \times 1$, the measurement probability distribution function (PDF) can be expressed as [5]

$$p(\mathbf{R}) = \frac{1}{(2\pi)^{N/2}|\mathbf{C_v}|^{1/2}} \exp^{[-1/2(\mathbf{R}-\mathbf{H}(\alpha)\mathbf{x})^T\mathbf{C_v}^{-1}(\mathbf{R}-\mathbf{H}(\alpha)\mathbf{x})]}. \tag{105}$$

The maximum likelihood estimate of $\mathbf{x}$, $\hat{\mathbf{x}}_{\mathbf{ML}}$, is found by minimizing the quadratic function [5]

$$V(x) = (\mathbf{R} - \mathbf{H}(\alpha)\mathbf{x})^T\mathbf{C_v}^{-1}(\mathbf{R} - \mathbf{H}(\alpha)\mathbf{x}). \tag{106}$$

The first order necessary condition to optimize $V(\alpha)$ is that $\partial V(x)/\partial x = 0$. For symmetric $\mathbf{C_v^{-1}}$,

$$\frac{\partial V(\alpha)}{\partial x} = 2\mathbf{H}(\alpha)^T\mathbf{C_v}^{-1}(\mathbf{R} - \mathbf{H}(\alpha)\mathbf{x}) \tag{107}$$

Setting $\partial V(x)/\partial x$ equal to zero yields (independent of σ_v^2)

$$\mathbf{H}(\alpha)^T\mathbf{C}^{-1}R = \mathbf{H}(\alpha)^T\mathbf{C}^{-1}\mathbf{H}(\alpha)\hat{x}_{ML} \tag{108}$$

If $\mathbf{H}(\alpha)$ is of full rank and $N \leq M-1$, then

$$\hat{x}_{ML} = (\mathbf{H}(\alpha)\mathbf{C}^{-1}\mathbf{H}(\alpha))^{-1}\mathbf{H}(\alpha)^T\mathbf{C}^{-1}\mathbf{R} \tag{109}$$

Because $\mathbf{C}^{-1}$ is positive definite, the existence of $(\mathbf{H}(\alpha)\mathbf{C}^{-1}\mathbf{H}(\alpha))^{-1}$ is sufficient to guarantee that $\hat{x}_{ML}$ is a global (unconstrained) minimum. The MLE, $\hat{x}_{ML}$, has been shown to be the minimum variance unbiased (MVU) estimator as well as an efficient estimator by [5]. The PDF of $\hat{\mathbf{x}}_{\mathbf{ML}}$ is given as $N(\mathbf{x}, (\mathbf{H}(\alpha)^T\mathbf{C_v}^{-1}\mathbf{H}(\alpha))^{-1})$.

$\hat{\mathbf{x}}_{\mathbf{ML}}$, via $\mathbf{H}(\alpha)$, is actually a function of $\alpha_i \in (0, 1.5]$, which does introduce constraints to the problem. If the primary path signal is perfectly synchronized with the receiver generated replica, then $\alpha_i < 0$ is not physically possible and $\alpha_i > 1.5$ corresponds with a delayed signal which has no impact upon the measurements, $\mathbf{R}$ [19].

An optimization to find the minimizing set of $\alpha_i \in (0, 1.5]$ can be obtained by substituting $\hat{x}_{ML}$ into Equation 106 as follows (Note: for $\sigma_v^2 > 0$, minimizing $\overline{V}(\alpha) \equiv \sigma_v^2 V(\alpha)$ is equivalent to minimizing $V(\alpha)$.)

$$\begin{aligned}
\overline{V}(\alpha) &= (\mathbf{R} - \mathbf{H}\mathbf{x}^T)\mathbf{C}^{-1}(\mathbf{R} - \mathbf{H}\mathbf{x})\,\big|_{x=\hat{x}_{ML}(\alpha)} \\
&= (\mathbf{R}^T - \mathbf{R}^T\mathbf{C}^{-1}\mathbf{H}(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T)\mathbf{C}^{-1}(\mathbf{R} - \mathbf{H}(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{C}^{-1}\mathbf{R}) \\
&= \mathbf{R}^T\mathbf{C}^{-1}\mathbf{R} - \mathbf{R}^T\mathbf{C}^{-1}\mathbf{H}(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{C}^{-1}\mathbf{R} \\
&= (\mathbf{C}^{-1}\mathbf{R})^T[\mathbf{C} - \mathbf{H}(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T](\mathbf{C}^{-1}\mathbf{R}).
\end{aligned} \tag{110}$$

$\mathbf{C}$ and $\mathbf{R}$ are known and measured, respectively, and the vector of unknowns is $\alpha \equiv [\alpha_1\ \alpha_2 \cdots \alpha_N]^T$ of Equation 97, which is needed to specify $\mathbf{H}(\alpha)$ of Equation 113. A minimization of $\overline{V}(\alpha)$ yields $\hat{\alpha}_{ML}$ which, in turn, determines $\hat{x}_{ML}$ via Equation 109. Common numerical optimization techniques for minimizing Equation 110 are quasi-Newton methods such as sequential quadratic programming (SQP). There are risks associated with

these iterative techniques: the iteration may not converge, the iteration may converge to a local minimum as opposed to the global minimum, or the signal-to-noise ratio may be insufficient for meaningful results.

3.5.1 Analysis of MCEU for a Single Reflection. Returning to the case of one reflected signal, the signal entering the MCEU can be represented as

$$r(t) = x_0 c(t - \tau_o) + x_1 c(t - \tau_o - \alpha T_c) + n'(t) \tag{111}$$

where $x_0 = \sqrt{2P} a_0 cos(\theta_0 - \theta_3)$ and $x_1 = \sqrt{2P} a_1 cos(\theta_1 - \theta_3)$. Let $v_k(t)$ represent the noise of the k^{th} MCEU correlator. Assuming perfect code tracking, the correlator outputs are

$$R_k(t) = x_0 R_c(\beta_k) + x_1 R_c(\alpha_1 - \beta_k) + v_k(t) \tag{112}$$

for $k = 0, 1, \ldots, M-1$ which can be represented collectively by the linear data model $\mathbf{R} = \mathbf{H}(\alpha)\mathbf{x} + \mathbf{v}$, where $\mathbf{x} \equiv [x_0 \; x_1]^T$ (Equation 100 for N=1),

$$\mathbf{H}(\alpha) = \begin{bmatrix} R_c(\beta_0) & R_c(\alpha_1 - \beta_0) \\ R_c(\beta_1) & R_c(\alpha_1 - \beta_1) \\ \vdots & \vdots \\ R_c(\beta_{M-1}) & R_c(\alpha_1 - \beta_{M-1}) \end{bmatrix}, \tag{113}$$

$\mathbf{R}$ is given in Equation 98, and α is given in Equation 97 for $N = 1$. When $\hat{\tau}_o = \tau_o$, the first column of $\mathbf{H}(\alpha)$ represents the cross correlation with the direct signal and the second column represents the cross correlation with reflected signal.

3.5.1.1 Numerical Optimization. Given that the samples, $\mathbf{R}$, obtained at the output of the bank of correlators conform to the statistics of Equation 105, the cost function $V(\alpha)$ of Equation 110 can be minimized numerically. Quasi-Newton methods build up curvature information of $V(\alpha)$ at each iteration to formulate a quadratic model problem (provided in Matlab Optimization Toolbox) from which the optimal solution is determined when the partial derivatives of the quadratic function go to zero. By using the observed

behavior of the quadratic function to build up curvature information, an approximation of the minimum can be determined with much fewer operations than traditional Newton-type methods which proceed in the direction of descent of the quadratic function using a line search method to locate the minimum after several iterations. Sequential Quadratic Programming (SQP) minimizes Newton's method for constrained optimization using a quasi-Newton updating method to approximate the Lagrangian function of $V(\alpha)$.

3.5.1.2 Theoretical Analysis of $V(\alpha)$ for a Single Reflection. The success of any SQP optimization routine depends on the algorithm's ability to determine the curvature of the given quadratic function from which it can converge to the minimum. Convergence to global minimum of a quadratic function can become very difficult given a series of local minimums. Figure 26 provides $V(\alpha)$ versus all possible values of $\alpha \in [0, 1.5]$ for a given multipath signal with delay α. As shown in the figure, the absolute or global minimum is indeed at the location of the multipath delay. However, the curvature of $V(\alpha)$ varies through the range of αs. The changes of slope (i.e., the ridges) of $V(\alpha)$ occur at the location of the arms of the correlators, $\beta_k T_c$. In Figure 26 a bank of M=11 correlators spaced evenly at increments of 0.15 through out the range of α. In addition the change of slope that occurs at 0.05, it is seen that there are $M-1$ curvature changes that occur at the delays corresponding to correlator locations interior to (0,1.5). Figure 27 provides $\mathbf{V}(\alpha)$ versus $\alpha \in (0, 1.5]$ for M=16. Once again, there $M-1$ changes of curvature, at the location of the interior correlator arms.

Because the slope of function is not defined at the location of the ridges, convergence problems can occur, as previously described. An SQP algorithm could potentially estimate the curvature of the slope of $V(\alpha)$ incorrectly which would corrupt the estimate of the delay, $\hat{\alpha}$ of the reflected signal. As such, the correlator spacing, $\beta_k T_c$, and the search algorithm employed should be considered carefully.

A previous design incorporated correlators spaced apart by $0.1T_c$ through the range of possible $\alpha \in [0, 1.5]$, which resulted in a bank of 16 correlators [6]. We have empirically determined that implementing a bank of 11 correlators, evenly spaced apart by $0.15T_c$, improves the performance of Matlab's constr.m SQP algorithm. By reducing the number

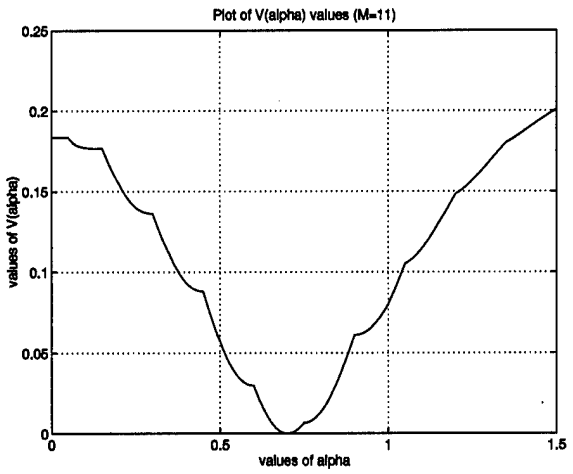


Figure 26 $V(\alpha)$ given $\alpha = 0.7$ where M=11 correlators are spaced at increments of 0.15 through the range [0,1.5].

of correlators, fewer changes of slope of the function, $V(\alpha)$, occur which improves the SQP algorithm's ability to converge; this simultaneously reduces the amount of hardware required. It was found that reducing the number of correlators further, led to degraded SQP algorithm performance.

Convergence problems have also been determined to occur for reflected signals with delays very close to the direct path signal. As the reflected signal becomes very close to the direct path signal, the columns of $\mathbf{H}(\alpha)$ become increasingly linearly dependent and $\mathbf{H}(\alpha)$ becomes rank deficient. As such, it is more difficult to numerically estimate the delays of reflected signals with delays very close to the direct path signal. Therefore, the SQP constraint function implemented was limited to estimating reflected signal delays of $\alpha \in [0.05, 1.5]$. Figure 28 provides $V(\alpha)$ versus α where the multipath delay is 0.05. The global minimum of the function is located at 0.05 which constitutes the edge of the search region. However, if allowed to search the region [0,0.05] the global minimum may not be located because the slope through this region is essentially zero. Figure 29 provides $V(\alpha)$ versus α where the multipath delay is 0.1. Once again, the global minimum of the function is located at the delay of the multipath signal.

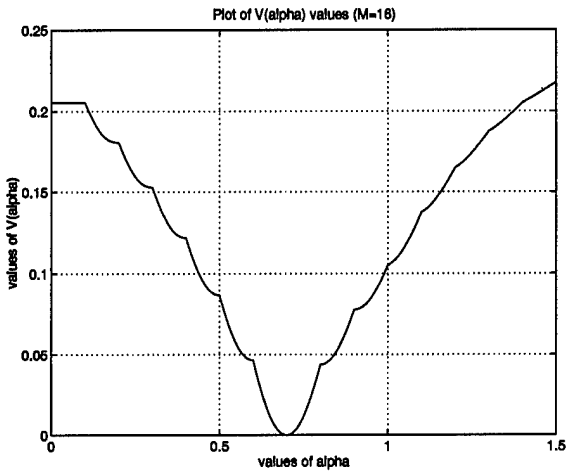


Figure 27 $V(\alpha)$ given $\alpha = 0.7$ where M=16 correlators are spaced at increments of 0.1 through the range [0,1.5].

Figures 30 and 31 provide $\mathbf{V}(\alpha)$ versus α for the region where the delays of the multipath signal are 1.5 and 1.4, respectively. As shown in the figures, convergence to the global minimum does not constitute a problem for these delays.

3.5.1.3 Search method for global minimum of $V(\alpha)$ for single reflection. In order to reduce convergence problems, a partitioned (1D) search technique can be used to find the global minimum of $V(\alpha)$. Since the undefined points/ridges are known to be located at the placement of the correlator arms, a search for the local minimum can be performed for the intervals between each of the arms. The local minimum for each interval can be determined by some numerical optimization routine and then compared to determine the global minimum. This method requires that the optimization routine be performed for each interval; however, convergence within each interval will require fewer iterations than a search across the entire range of α, since the quadratic function through each interval is smooth (with no undefined points/ridges). The global minimum could possibly be converged on more quickly by searching each interval using parallel processing such that the local minimums through the partitions of $V(\alpha)$ are determined concurrently. This approach can be generalized for more than a single reflection.

Table 3 provides the results of the search algorithm employed using the *constr.m* function in Matlab for a single multipath delay and M=11 correlators. The results provided

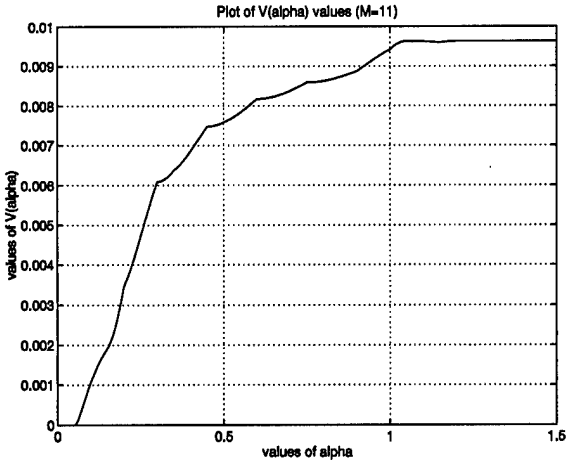


Figure 28 $V(\alpha)$ given $\alpha = 0.05$ where M=11 correlators are spaced at increments of 0.15 through the range [0,1.5]..

in the table, for α in uniformly spaced increments of 0.05, indicate that the algorithm was able to estimate each of the signal parameters with an accuracy $\geq 99.95\%$.

3.5.2 Analysis of MCEU for Multiple Reflections. For the case of two reflected signals, $\mathbf{x} = [x_0\ x_1\ x_2]^T$ and $\mathbf{H}(\alpha)$ is an $M \times 3$ matrix with the first column representing the cross correlation with the direct signal and the second and third columns representing the cross correlation with each of the reflected signals. Figure 32 provides $V(\alpha)$ versus all possible values of α_1 and α_2 through the range α_1,$\alpha_2 \in [0, 1.5]$. The ridges or changes of curvature are apparent along the direction of α_1 and α_2. The figure, as expected, has symmetry about the diagonal of $\alpha_1 = \alpha_2$; the elements of $V(\alpha)$ are undefined at these locations.[3] The physical interpretation of $\alpha_1 = \alpha_2$ is that only a single reflection exists.

Figure 33 represents a slice taken at the known location, $\alpha = 0.2$, of one of the given multipath delays. This figure exhibits the same characteristics as the case for one reflection.[4] Ridges occur at the locations of the arms, $\beta_k T_c$, as previously described. In addition, the absolute or global minimum is located at the point of the other given multipath delay.

[3]The diagonal elements of $V(\alpha)$ have been set to some arbitrary value in the 2D figures presented although they are undefined because $\mathbf{H}(\alpha)$ is rank deficient when $\alpha_1 = \alpha_2$.

[4]The point at 0.2 should be ignored since it as the location of the diagonal element.

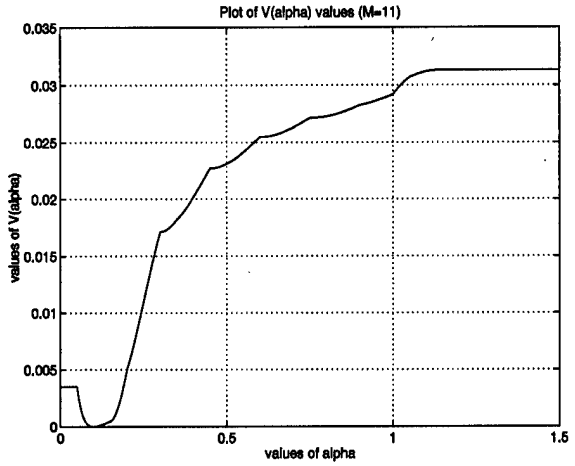


Figure 29 $V(\alpha)$ given $\alpha = 0.1$ where M=11 correlators are spaced at increments of 0.15 through the range [0,1.5]..

Figure 34 provides $\mathbf{V}(\alpha)$ versus $\alpha_1, \alpha_2 \in (0, 1.5]$ given two multipath delays which differ from those used in Figure 32. Characteristics similar to those of Figure 32, such as the ridges, can be recognized. Figure 35 provides a close-up of the location of the given multipath delays of Figure 34. As shown in the figure, the global minimum is indeed located at $\alpha_1 = 0.5$ and $\alpha_2 = 0.9$. Due to symmetry of $V(\alpha)$ about the diagonal, a like minimum exists at $\alpha_1 = 0.9$ and $\alpha_2 = 0.5$.

Figure 36 provides $\mathbf{V}(\alpha)$ versus $\alpha_1, \alpha_2 \in (0, 1.5]$ given two multipath signals with closely spaced delays, $\alpha_1 = 0.9$ and $\alpha_2 = 1.0$. As shown in the figure, $\mathbf{V}(\alpha)$ becomes very flat near the location of the multipath delays, which may cause difficulties for an SQP algorithm to converge to the global minimum. In fact, without a search routine, the SQP algorithm implemented in Matlab performed poorly with multipath delays spaced relatively close together. As such, a search technique may provide increased performance.

3.5.2.1 Search method for global minimum of $V(\alpha)$ for multiple reflections. The search technique applied for a single reflection may be implemented for multiple reflections. Implementing a search technique for two reflected signals requires a grid (2D) search technique to find the global minimum. Since $V(\alpha)$ is symmetric about the diagonal, only grid regions on one side of the diagonal need to be searched.

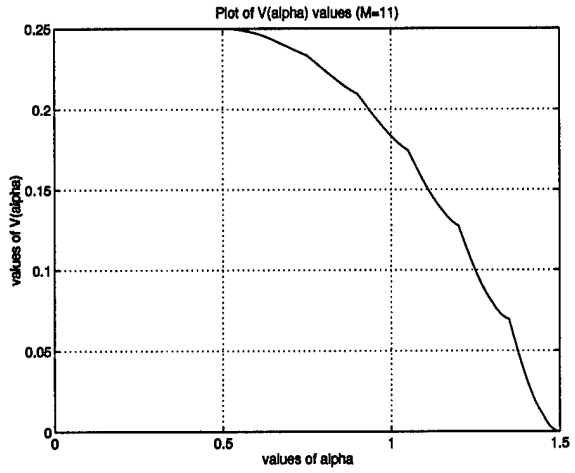


Figure 30 $V(\alpha)$ given $\alpha = 1.5$ where M=11 correlators are spaced at increments of 0.15 through the range [0,1.5].

This symmetry holds true for more than two reflected signals as well, but requires significantly increased computational burden as the number of multipath signals increases. The number of search regions required for a given set of multipath signals can be expressed as a combination

$$\begin{pmatrix} n+k-1 \\ k \end{pmatrix} \tag{114}$$

where k equals the number of multipath dimensions (i.e., k-1 reflected signals), n equals the number of search intervals (in one dimension) between correlator arms, and $n \geq k$.

For two reflected signals with M=11 correlators, the number of search regions is

$$\begin{pmatrix} 10+2-1 \\ 2 \end{pmatrix} = \begin{pmatrix} 11 \\ 2 \end{pmatrix} = \frac{11!}{2!9!} = 55. \tag{115}$$

These regions were searched to locate the global minimum for the given set of multipath delays, α_1 and α_2. In order to test the performance of the SQP search algorithm the true α_1 and α_2 were varied by increments of 0.1 through the range [.1,1.5] which provided 105 test cases. Of the 105 cases, 88 or 83.8% of the cases resulted in estimation accuracy of $\geq$ 99.9% for the parameters $\hat{\alpha}_1$, $\hat{\alpha}_2$, $\hat{x}_0$, $\hat{x}_1$, and $\hat{x}_2$. The results of the remaining 17

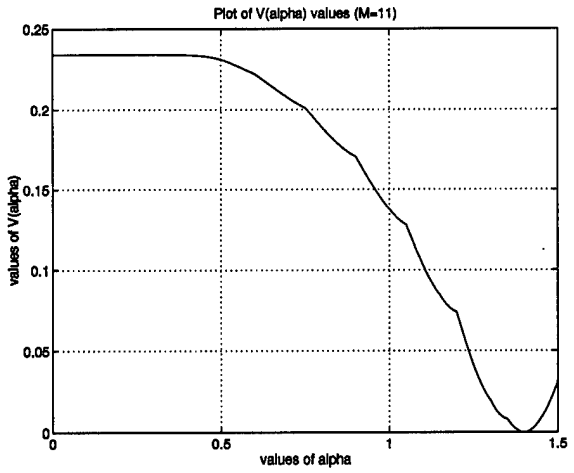


Figure 31 $V(\alpha)$ given $\alpha = 1.4$ where M=11 correlators are spaced at increments of 0.15 through the range [0,1.5].

test cases are provided in Table 4. All of the multipath delays presented in Table 4 are spaced ≤ 0.2 apart with the worst results occurring for delays spaced $= 0.1$. Even so, the estimates of the delays were within 0.1 chips of the true delay. It should also be noted that for six of the cases, estimates of x_1 and x_2 were very high, ≥ 10. This information could be used in an algorithm to indicate very poor estimates.

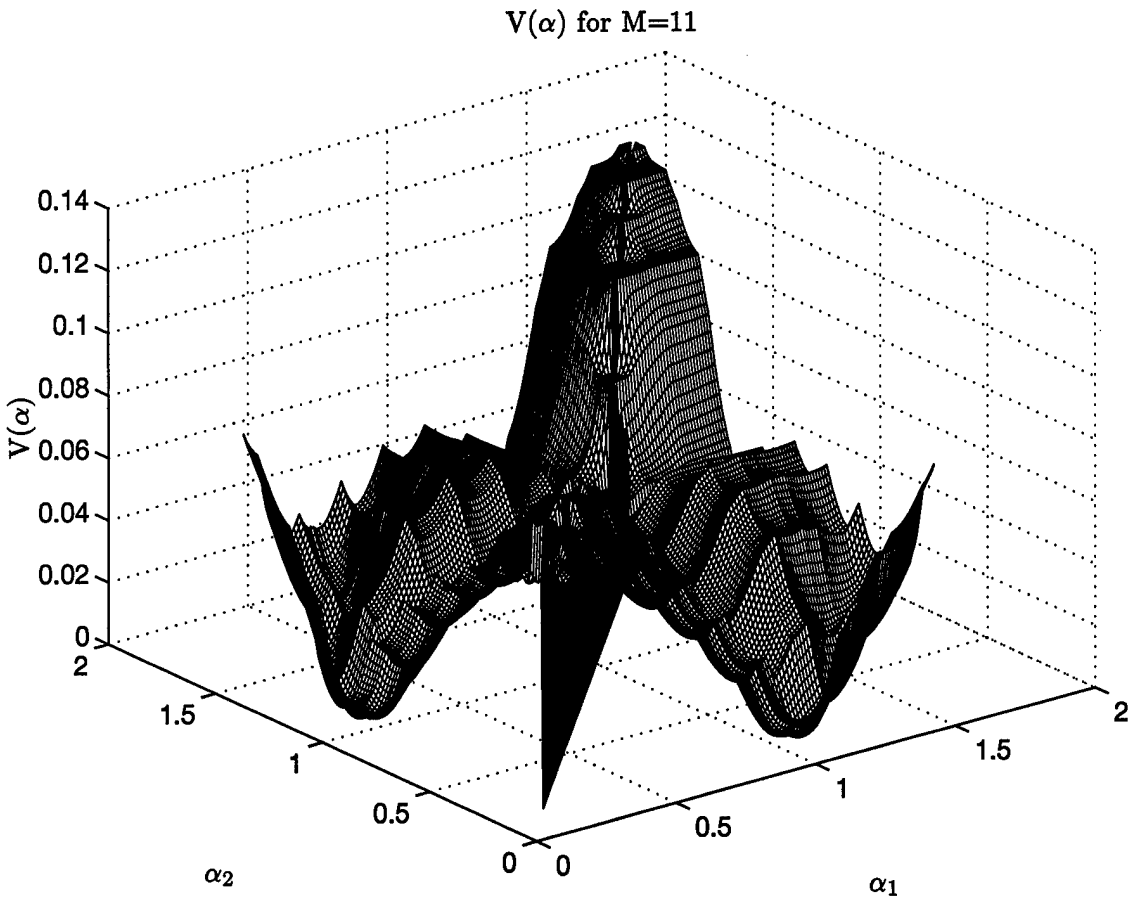


Figure 32 $V(\alpha)$ given $\alpha_1 = 0.2$ and $\alpha_2 = 1.1$.

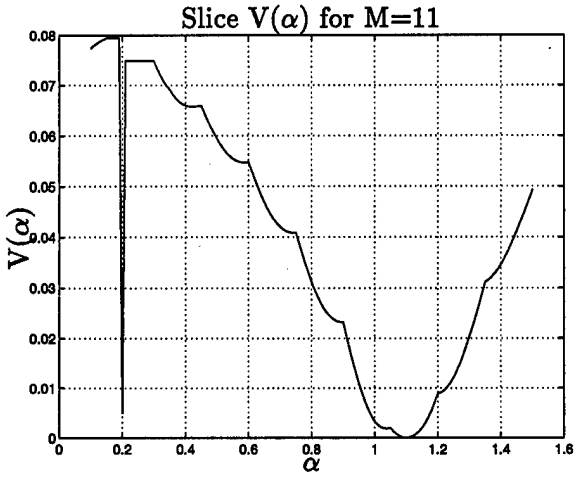


Figure 33 Slice of $\mathbf{V}(\alpha$ given $\alpha_1 = 0.2$ and $\alpha_2 = 1.1$.

Table 3 Table of true values of α, x_0, and x_1 with corresponding estimates

α	x_0	x_1	$\hat{\alpha}$	$\hat{x}_0$	$\hat{x}_1$
0.0500	1.0000	-0.5000	0.0500	1.0000	-0.5000
0.1000	1.0000	0.5000	0.1000	1.0000	0.5000
0.1500	1.0000	-0.5000	0.1500	1.0000	-0.5000
0.2000	1.0000	0.5000	0.2000	1.0000	0.5000
0.2500	1.0000	-0.5000	0.2500	1.0000	-0.5000
0.3000	1.0000	0.5000	0.3000	1.0000	0.5000
0.3500	1.0000	-0.5000	0.3500	1.0000	-0.5000
0.4000	1.0000	0.5000	0.4000	1.0000	0.5000
0.4500	1.0000	-0.5000	0.4500	1.0000	-0.5000
0.5000	1.0000	0.5000	0.5000	1.0000	0.5000
0.5500	1.0000	-0.5000	0.5500	1.0000	-0.5000
0.6000	1.0000	0.5000	0.6000	1.0000	0.5000
0.6500	1.0000	-0.5000	0.6500	1.0000	-0.5000
0.7000	1.0000	0.5000	0.7000	1.0000	0.5000
0.7500	1.0000	-0.5000	0.7500	1.0000	-0.5000
0.8000	1.0000	0.5000	0.8000	1.0000	0.5000
0.8500	1.0000	-0.5000	0.8500	1.0000	-0.5000
0.9000	1.0000	0.5000	0.9000	1.0000	0.5000
0.9500	1.0000	-0.5000	0.9500	1.0000	-0.5000
1.0000	1.0000	0.5000	1.0000	1.0000	0.5000
1.0500	1.0000	-0.5000	1.0500	1.0000	-0.5000
1.1000	1.0000	0.5000	1.1000	1.0000	0.5000
1.1500	1.0000	-0.5000	1.1500	1.0000	-0.5000
1.2000	1.0000	0.5000	1.2000	1.0000	0.5000
1.2500	1.0000	-0.5000	1.2500	1.0000	-0.5000
1.3000	1.0000	0.5000	1.3000	1.0000	0.5000
1.3500	1.0000	-0.5000	1.3500	1.0000	-0.5000
1.4000	1.0000	0.5000	1.4000	1.0000	0.5000
1.4500	1.0000	-0.5000	1.4500	1.0000	-0.5000
1.5000	1.0000	0.5000	1.5000	1.0000	0.5000

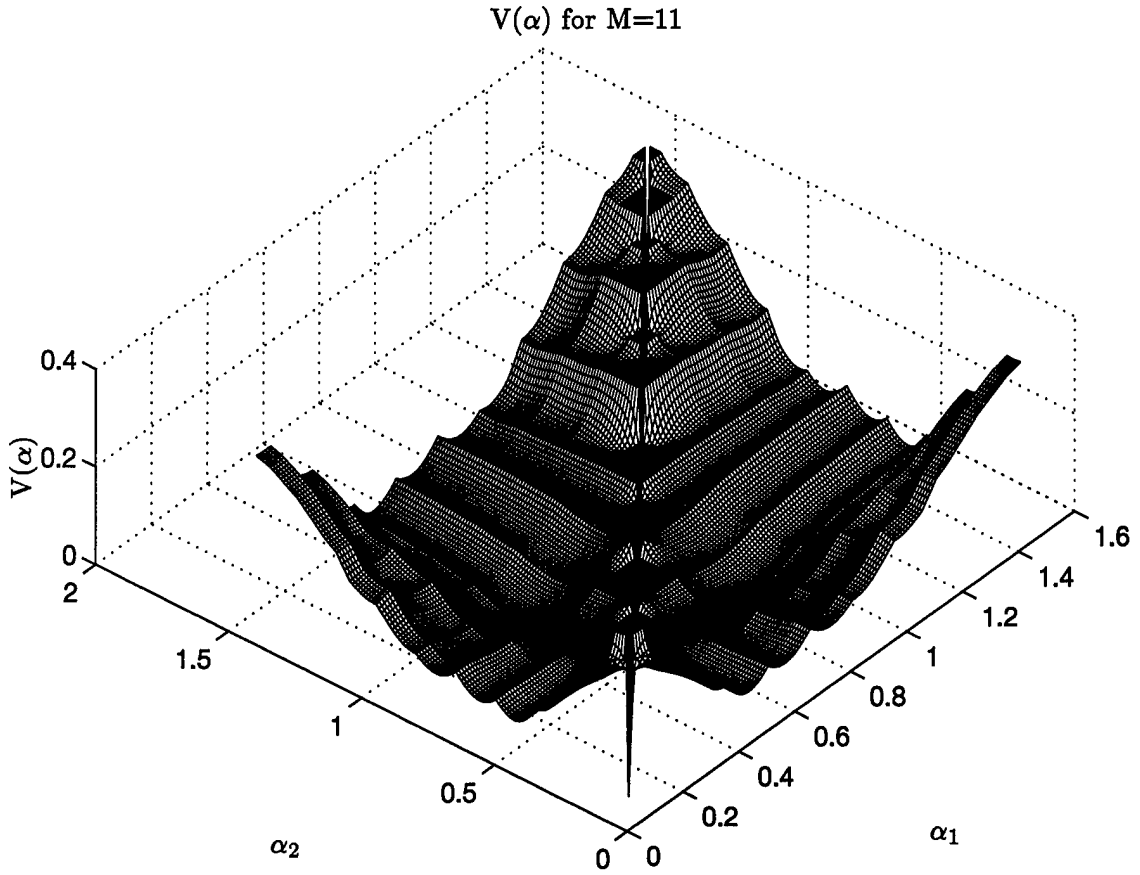


Figure 34 $V(\alpha)$ given $\alpha_1 = 0.5$ and $\alpha_2 = 0.9$.

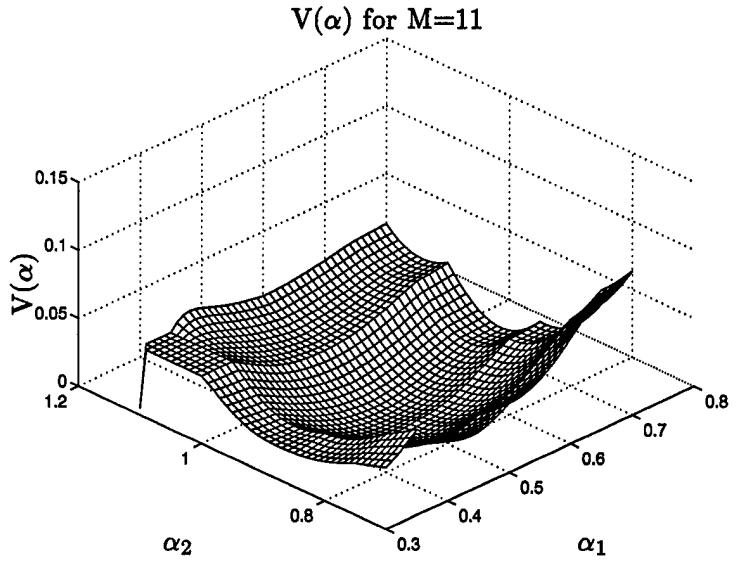


Figure 35 $V(\alpha)$ given $\alpha_1 = 0.5$ and $\alpha_2 = 0.9$.

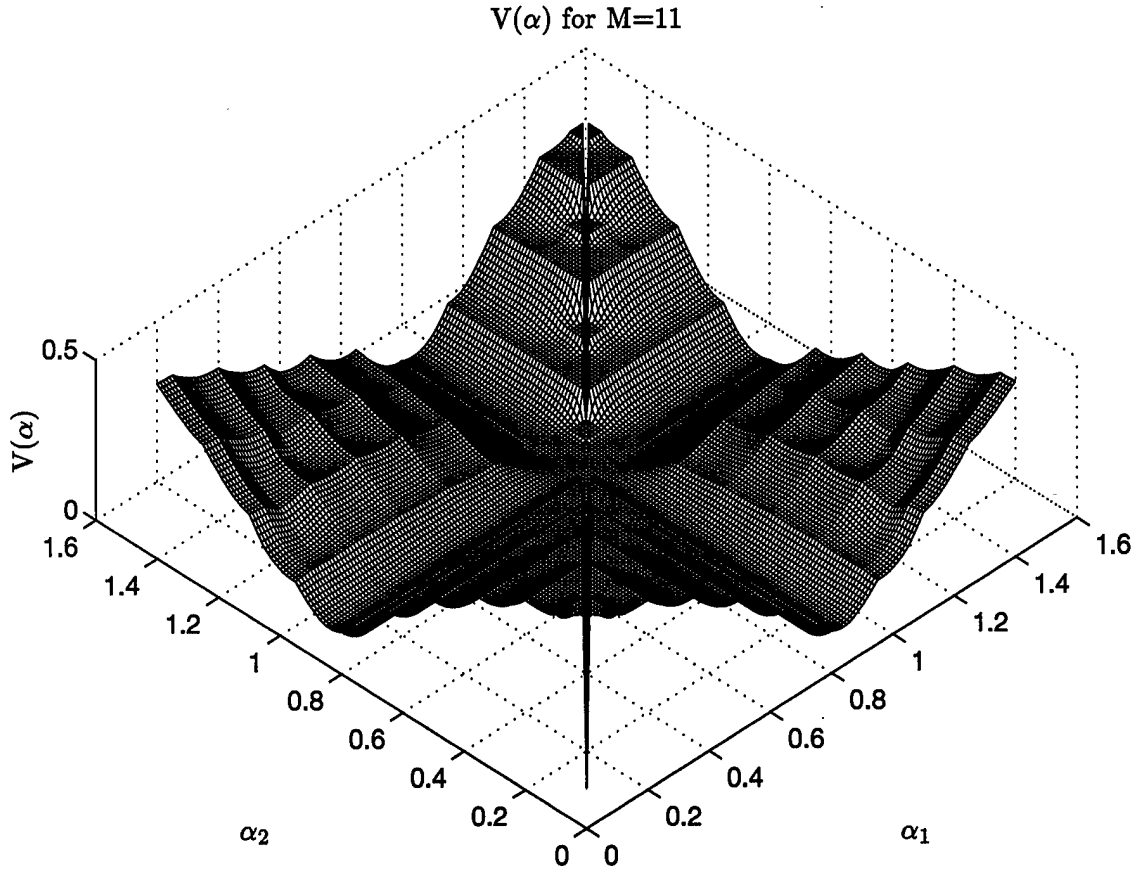


Figure 36 $V(\alpha)$ given $\alpha_1 = 0.9$ and $\alpha_2 = 1.0$.

Table 4 Table of true values of α_1, α_2, x_0, x_1, and x_2 with corresponding estimates

α_1	α_2	x_0	x_1	x_2	$\hat{\alpha}_1$	$\hat{\alpha}_2$	$\hat{x}_0$	$\hat{x}_1$	$\hat{x}_2$
0.2000	0.3000	1.0000	0.5000	0.3000	0.2000	0.2010	1.0000	-29.0710	29.8710
0.2000	0.4000	1.0000	0.5000	0.3000	0.2001	0.4001	1.0000	0.5005	0.2995
0.3000	0.4000	1.0000	0.5000	0.3000	0.3017	0.4031	1.0000	0.5175	0.2825
0.4000	0.5000	1.0000	0.5000	0.3000	0.3755	0.4823	1.0000	0.3354	0.4646
0.5000	0.6000	1.0000	0.5000	0.3000	0.5000	0.5011	1.0000	-27.1234	27.9234
0.6000	0.7000	1.0000	0.5000	0.3000	0.6500	0.6510	1.0000	10.3104	-9.5104
0.6000	0.8000	1.0000	0.5000	0.3000	0.6083	0.8054	1.0000	0.5292	0.2708
0.7000	0.8000	1.0000	0.5000	0.3000	0.6697	0.7807	1.0000	0.3113	0.4886
0.7000	0.9000	1.0000	0.5000	0.3000	0.6621	0.8373	1.0000	0.2846	0.5154
0.8000	0.9000	1.0000	0.5000	0.3000	0.8000	0.8011	1.0000	-27.1771	27.9771
0.8000	1.0000	1.0000	0.5000	0.3000	0.7932	0.9904	1.0000	0.4681	0.3319
0.9000	1.0000	1.0000	0.5000	0.3000	0.9499	0.9509	1.0000	10.1132	-9.3132
1.0000	1.1000	1.0000	0.5000	0.3000	1.0031	1.1062	1.0000	0.5332	0.2668
1.1000	1.2000	1.0000	0.5000	0.3000	1.0500	1.1773	1.0000	0.2500	0.5500
1.2000	1.3000	1.0000	0.5000	0.3000	1.2511	1.2521	1.0000	11.5485	-10.7485
1.3000	1.4000	1.0000	0.5000	0.3000	1.2000	1.3714	1.0000	0.1659	0.6325
1.4000	1.5000	1.0000	0.5000	0.3000	1.3862	1.4867	1.0000	0.3917	0.4083

IV. Analysis

4.1 Overview

The theory of operations of the eMRDLL and the NCDLL provided in Chapters 2 and 3 describe the expected performance characteristics of these designs. However, ideal conditions were assumed in the derivations of the principle concepts of these designs. As such, analysis of these designs in a realistic environment is necessary in order to test their effectiveness. Computer modeled simulations provide this realistic environment.

This Chapter analyzes the operation of the eMRDLL and the NCDLL in a simulated environment. The eMRDLL is used as the platform to test the effectiveness of ML estimation in a GPS environment. The NCDLL, with correlator spacings of $\Delta = 1.0$ and $\Delta = 0.1$, serves as the baseline design for which the performance of MRDLL can be compared.

Simulations were conducted using a graphical environment provided by the Simulink Toolbox, version 1.3 which requires the use of Matlab, version 4.2c to provide computational support. In addition, functions from the Signal Processing Toolbox, the Optimization Toolbox, and the Communications Toolbox were used. All software packages used in these simulations were provided by The MathWorks, Inc. of Natick, Massachusetts. A description of the models developed is provided in Appendix *A*. Simulations were performed on the Sun Workstations provided at the Air Force Institute of Technology (AFIT).

4.2 Simulations Performed

Analysis of five simulation scenarios are presented in order to characterize the performance characteristics of eMRDLL.

- Simulation # 1: Performance analysis is conducted on the MCEU (without AWGN) for various multipath delays $\alpha \in (0, 1.5]$. Perfect synchronization is assumed so that the MCEU is isolated. Analysis of the MCEU is provided with and without the baseband mixing process implemented.

- Simulation # 2: Performance analysis is conducted on the MCEU for various multipath delays $\alpha \in (0, 1.5]$ for a range of SNR$\in [-35, 30]$ dB. Perfect synchronization is assumed so that the MCEU is isolated.

- Simulation # 3: Closed loop code phase tracking analysis is performed (without AWGN) for the eMRDLL versus the NCDLL ($\Delta = 1.0$ and $\Delta = 0.1$) for various multipath delays $\alpha \in (0, 1.5]$.

- Simulation # 4: Closed loop code phase tracking analysis is performed for the eMRDLL for various multipath delays $\alpha \in (0, 1.5]$ for a range of SNR$\in [-35, 30]$ dB.

- Simulation # 5: Performance analysis is conducted on the MCEU for time varying multipath delays with and without AWGN. Perfect synchronization is assumed so that the MCEU is isolated.

4.3 Simulation Parameters

In order to provide an accurate analysis of the eMRDLL and the NCDLL, signal parameters were chosen so as to best emulate the functionality of a receiver in a GPS environment. Software limitations, however, existed such that many parameters found in GPS applications had to be scaled. The following sections outline the parameters used for simulations conducted as well as the rationale for choosing these parameters.

In Choosing the simulation parameters properly, the following were considered as constraints:

- In a simulation environment, such as Simulink, the sampling rate, f_s, must be finite. Increasing the sampling rate increases simulation run-times, so there exists a trade-off between computational burden and aliasing considerations.

- The code rate, $1/T_c$ must be chosen high enough to ensure proper operation of receiver components such as the VCC; for example, a negative input into the VCC with a code rate that is very low may drive the frequency of the VCC to zero at which point it cannot be lowered further (i.e, cannot produce negative frequency).

- Filters with very narrow bandwidths, implemented in Simulink, require a carrier frequency to code rate ratio, $f_o : 1/T_c$, not to exceed 10:1 in order to ensure proper performance. Parameters chosen which exceed this ratio lead to amplification of the signal within the filter's passband.

4.3.1 Spreading Code Chip Rate, $1/T_c$. The determination of the chip rate, $1/T_c$, was a fundamental consideration in designing the models of the eMRDLL and the NCDLL. A chip rate of $1/T_c = 100$ *chips/sec* (or *Hz*) was chosen to ensure proper operation of receiver components such as the VCC. The clock rate of the VCC had to be considered such that it would never reach zero (the lower limit of the VCC dynamic range) based on the error signal coming from the loop filter, F(s). Depending on the gain of the system, a chip rate less than 100 *chips/sec* could result in the VCC being driven to zero. A higher chip rate also allowed for more flexibility in choosing the loop bandwidth, B_L, for the NCDLL (i.e., smaller loop bandwidths could be chosen for higher chip rates).

4.3.2 Carrier Frequency, f_o . In order to ensure that the filters performed properly, a carrier frequency of $f_o = 1000\ Hz$ was chosen so as to maintain a carrier to code chip rate ratio of 10:1. This also helped keep the sampling frequency, f_s, lower which was necessary to maintain reasonable simulation run times.

There was, however, a trade-off in choosing a carrier frequency which was relatively close to the code rate. As will be shown in the following sections, higher frequency harmonics, which resulted from the pre-correlation mixing process, were introduced into the baseband signal entering the eMRDLL.

4.3.3 Sampling Frequency, f_s. The sampling frequency, f_s, was chosen relative to the carrier frequency, f_o, and the code chip rate, $1/T_c$. A chip resolution of 100 *samples/chip* was desired in order to be able to consider a broad range of multipath delays. Resolution less than $.01T_c$ would have resulted in a more limited range of possible multipath delays. Also, at least 10 samples per carrier cycle were desired for Nyquist rate considerations regarding representation of the received multipath signal. In order to maintain reasonable simulation run times and meet the sampling requirements described above, f_s was chosen as $10,000\ Hz$. This resulted in a simulation integration step size of $1/f_s = 0.0001\ sec$.

4.3.4 C/A (DS/SS) Code Length, N. GPS satellites transmit a Gold Code[1] with maximum length pseudo-noise (PN) characteristics. As stated in Chapter 1, the C/A spreading code has $N = 1023$ chips per code cycle. Preliminary simulations indicated that the response time of using an $N = 1023$ chip sequence was too lengthy. As such, a maximum length PN sequence of $N = 2^m - 1 = 63$ *chips*, with code shift register order of $m = 6$ was used, as described in Appendix A. Choosing $N = 63$ for the simulations still permits the use of the large N code correlation approximation of Equation 8 with the added benefit of increasing simulation performance.

4.3.5 BPF and LPF Bandwidths, B. As stated in Chapter 2, the bandwidths of the filters in the 'early'-'late' gate branches should be chosen on the order of the data rate. In GPS, this corresponds to 50 Hz which is a fraction of the spreading code chip rate of 1.023 MHz. Trying to set the filter bandwidths of the models used in Simulink with the same proportion of that in GPS would have resulted in unrealizable constraints placed on the filters. van Nee states that, for GPS, these pre-detection bandwidths are typically in the range of 1 kHz down to the *two-sided* null-to-null data bandwidth of 100 Hz [19]. The integration period determined from the pre-detection bandwidth of 1 kHz would be equivalent to $1/NT_c$. This is more reasonable for a simulation environment than a bandwidth chosen on the order of the data rate.

The code correlation function, $R_c(\tau)$, of Equation 7, can be implemented with a code multiplier followed by either a BPF with bandwidth $B = 2/NT_c$ Hz or LPF with bandwidth $B = 1/NT_c$, where either filter acts as an integrator. As such, the LPFs used in eMRDLL were set at $B = 1/NT_c = 1.6$ Hz. In the NCDLL models, the BPF bandwidths could not be set at $B = 2/NT_c = 3.2$ Hz due to filter bandwidth constraints encountered with Simulink;[2] consequently, the bandwidths of the BPFs had to be set to a slightly higher value of $B = 4$ Hz.

[1] The gold code represents the modulo-two addition of two maximum length sequences generated by 10 bit registers.

[2] Filter bandwidths that were too narrow resulted in signals that were improperly filtered.

4.3.6 Loop Filter Bandwidth, B_L. In most GPS receivers, the loop integration time is roughly the reciprocal of the code-tracking loop bandwidth. Bandwidths of code tracking loops can vary between 0.02 Hz and 1 Hz [19,21]. In conducting the simulations, the loop filter bandwidth could not be scaled lower, because DC terms would have been attenuated in the same manner as higher frequency components. The first order active lead-lag filter should integrate DC terms; therefore, a loop filter bandwidth was chosen within the range of an actual GPS receiver. A loop filter bandwidth of $B_L = 0.1\ Hz$ ($\omega_n = 0.06\pi\ rad/sec$ and $\zeta = 1/\sqrt{2}$ in by Equation 89) provided the best performance for the NCDLL model implemented in Simulink. The parameters $\omega_n = 0.06\pi\ rad/sec$ and $\zeta = 1/\sqrt{2}$ were left unchanged, relative to the NCDLL, in designing the ALC of the eMRDLL; however, the loop bandwidth does vary with $\hat{A}$ due to the presence of the ALC.

4.3.7 Signal to Noise Ratio, SNR . SNR was computed via the ratio of the power of the transmitted signal, Pa_0^2, to the noise power, N, which is determined by the variance of the noise, σ^2. In using the AWGN noise block, provided by the Communications Toolbox for use in Simulink, it was discovered that the sampled noise output had a different variance over a period of 100,001 samples than what was specified as the input variance. This was true for all variance inputs used in the simulations. As such the SNR levels specified did not correspond with calculated SNR levels. Table 5 provides the input SNR versus the calculated SNR used in the simulations discussed in this chapter.

Table 5 Specified SNR versus calculated SNR

Specified SNR (dB)	-35	-23	-13	-6	0	5	10	15	20	25	30
Calculated SNR (dB)	-19	-13	-8	-4.5	-1.5	1	3.5	6	8.5	11	13.5

4.3.8 Simulation Parameter Summary . The following provides a list of all simulation parameters used in the following analysis. Parameters described below which were not discussed in the previous sections were based on the designer's preference.

- Code chip rate: $1/T_c = 100\ chips/sec$ (or 100 Hz)
- Carrier frequency: $f_o = 1000\ Hz$

- Simulation sampling frequency: $f_s = 10000\ Hz$
- Code length: $N = 63$
- BPF bandwidth (NCDLL): $B = 4\ Hz$
- BPF type: 2^{nd} Order Butterworth
- LPF bandwidth (eMRDLL and NCDLL): $B = 1.6\ Hz$
- LPF type: 5^{th} Order Butterworth
- Loop filter bandwidth: $0.1\ Hz$
- Transmitted signal power: $P = 1/2$
- Signal amplitudes: $a_0 = 1.0$ and $a_1 = 0.5$
- Direct-path propagation delay: $\tau_o = 0$
- 'Early-late' normalized correlator spacing (NCDLL): $\Delta = 1.0$, $\Delta = 0.1$
- 'Early-late' normalized correlator spacing (eMRDLL): $\Delta = 0.1$
- Loop natural frequency: $\omega_n \approx 0.06\pi\ rad/sec$
- Loop damping factor: $\zeta = 1/\sqrt{2}$
- Simulink (relative) SNR levels: -35 dB to 30 dB

4.4 Simulation # 1: MCEU Noise Free Performance

The code phase tracking accuracy of the eMRDLL is based largely on the performance of the MCEU. As such, the MCEU was analyzed to characterize its behavior and performance characteristics. The analysis in the following section was performed assuming that the MCTL was perfectly synchronized at all times, which isolated the performance of the MCEU. This analysis was performed without AWGN; a noise-free case.

4.4.1 MCEU Performance (without baseband mixing operation). As discussed in section 4.3.2, the choice of the carrier frequency, f_o, resulted in some harmonic spillage into the spectrum of the baseband signal. The objective of this analysis is to characterize the

performance of the MCEU for the case where the spreading code is already at baseband; thus, removing the effects of the frequency conversion from IF to baseband.

In implementing the MCEU, the rate at which the MCEU provides estimates to the MCTL and ALC must be chosen. For a system operating at real-time, it would be undesirable and unnecessary to estimate the multipath parameters at a sampling rate higher than the code rate; a practical system requires estimates to be provided at most once every code period, NT_c, depending on the application of the receiver.[3] The MCEU was designed to provide estimates to the MCEU once every code period, $NT_c = 0.63$ *sec.* Since filters are being used as integrators, the absolute time at which estimates are sampled is very important. To demonstrate the importance of sampling time, the estimates from the MCEU were calculated every one-tenth of a cycle, 0.063 *sec.* Figures 37 and 38 provide plots of the estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$. Each figure illustrates the true values of α, x_0, and x_1 as well as estimates of these parameters at sampling intervals of 0.63 *sec* and 0.063 *sec.* The choice of where sampling occurred was empirically determined based on the response of the LPFs implemented.

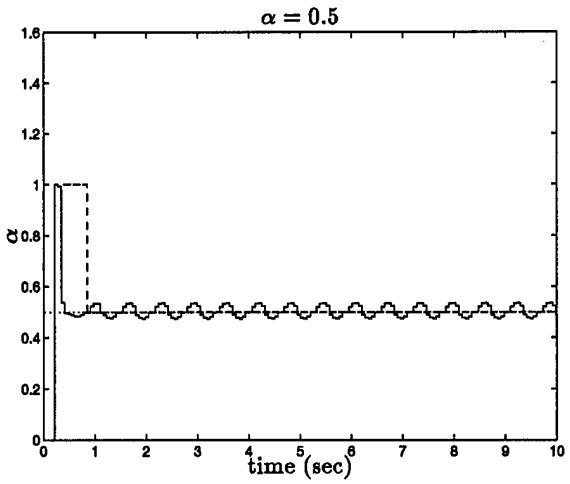
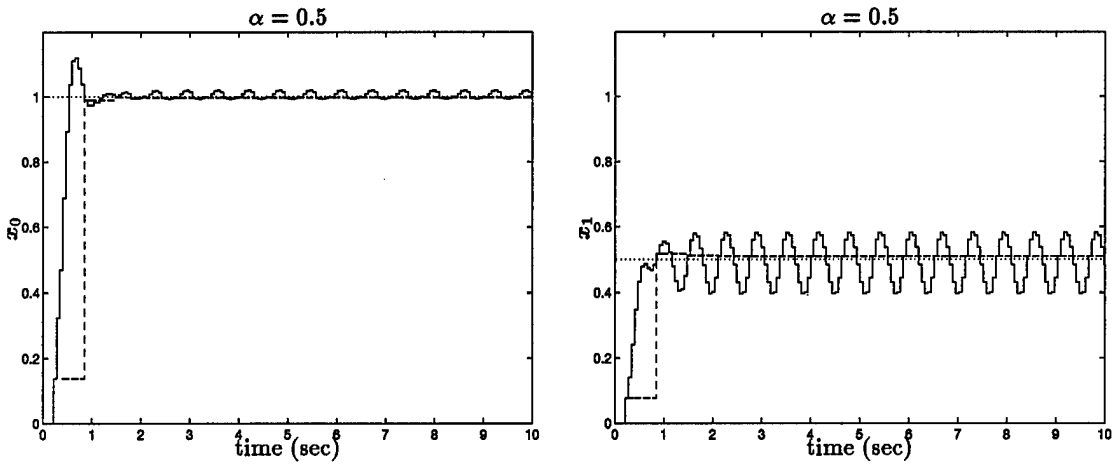


Figure 37 MCEU performance without baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of α, '—' represents MCEU estimates of α sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of α sampled once every cycle.

[3]Receivers operating in environments with slowly time varying multipath would not need to update signal estimates as frequently as receivers operating in a dynamic multipath environment.



a) MCEU estimation of x_0 b) MCEU estimation of x_1

Figure 38 MCEU performance without baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of x_0 and x_1, '—' represents MCEU estimates of x_0 and x_1 sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of x_0 and x_1 sampled once every cycle.

From the figures, the sampling interval of 0.63 *sec* is evident as the period of the steady-state response of the estimates provided by the MCEU for $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$; these oscillations are caused by the LPFs used in the bank of correlators. The code autocorrelation, $R(\tau)$, function provided in Equation 7, indicates that $R(\tau)$ is time averaged over a single period. One can consider the operation of $R(\tau)$ over time as a moving window integrator that calculates the area of the received spreading code coinciding with a locally generated code replica, delayed by $\beta_k T_c$, for each corresponding correlator arm. Assuming perfect synchronization, $R(\tau)$ would remain constant for an ideal integrator; however, the LPFs used in the correlator arms are not ideal. The outputs of the correlator arms vary with time as shown in the figures. This results in the MCEU producing estimates that also vary with time. As it was noted, the responses of the filters and thus the estimates of the MCEU are periodic with period equal to the length of the code period, NT_c; consequently, the figures illustrate that the MCEU estimates indeed have a period of $NT_c = 0.63$ *sec*.

Based on the results provided above, one can tune estimator performance by adjusting the epoch at which the MCEU periodic samples are collected; this can be established a priori based on the response characteristics of the particular LPF implemented.

Figures 37 and 38 indicate that the the sampling time implemented for the sampling rate of 1/.63, was chosen very well since the estimates, $\hat{\alpha}$ and $\hat{x}_0$, are very near to the true values, α and x_0. The estimate $\hat{x}_1$, however, is not as accurate as the estimates of the other two parameters.

The sensitivity of $\hat{x}_1$ can best be described by considering that $\hat{\alpha} \neq \alpha$ causes $\hat{\phi}_e \neq \phi_e$ and the baseband amplitudes of x_0 and x_1 deviate as presented in Figure 39; this figure provides the expected values of x_0 and x_1 for signal amplitudes of $a_0 = 1.0$ and $a_1 = .5$ as α is varied through (0,1.5]. It is interesting to note that the range of amplitude of the reflected component is much larger than that of x_0 and that the envelope amplitude of x_0 exponentially decreases to zero as α increases from zero. Figure 39 thus illustrates how the estimates $\hat{x}_0$ and $\hat{x}_1$ can be expected to oscillate with oscillating values of $\hat{\alpha}$.

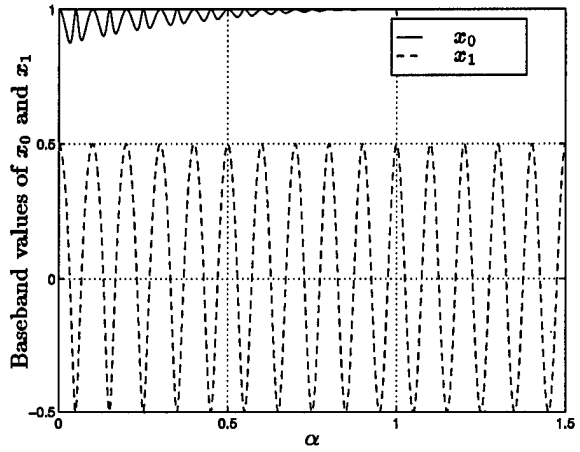


Figure 39 Baseband values of x_0 and x_1 for $P = 1/2$ and $f_o T_c = 10$.

4.4.2 MCEU Performance (with baseband mixing operation). Now that the baseline performance of the MCEU has been characterized, the effects of the baseband conversion process on the MCEU can be analyzed. Figures 40 through 45 illustrate the performance of the MCEU for three cases, where $\alpha = 0.1, 0.5$, and 1.3. As with the previous section, the estimates were calculated every 0.063 *sec* as well as estimates at sampling intervals of 0.63 *sec*. Figures are provided for $\hat{\alpha}$ and $\hat{x}_0$, $\hat{x}_1$ for each of the multipath cases examined in this section.

4.4.2.1 MCEU Performance for $\alpha = 0.1$. The estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$, provided in Figures 40 and 41, illustrate the sensitivity caused by the LPF responses at small delays of α, such as $\alpha = 0.1$. The estimates provided by the MCEU cover the full range of possible values of $\hat{\alpha}$. The estimates are also periodic in nature with period NT_c as previously described. Although the samples taken every 0.63 seconds provide an estimate $\hat{\alpha} = 0.15$, most 0.063 *sec* estimates are far away from the true $\alpha = 0.1$. The MCEU estimates $\hat{x}_0$ and $\hat{x}_1$ also vary greatly based on varying $\hat{\alpha}$ estimates. As presented in Figure 39, both x_0 and x_1 are very sensitive for variations of α near the direct-path signal. Figure 41b also shows that as $\hat{\alpha}$ varies, $\hat{x}_1$ may become negative.

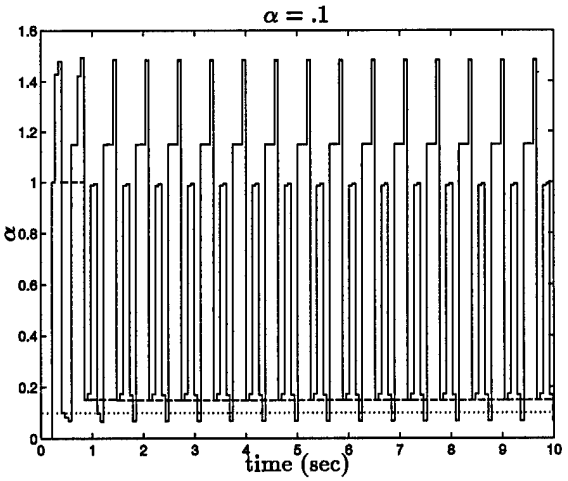
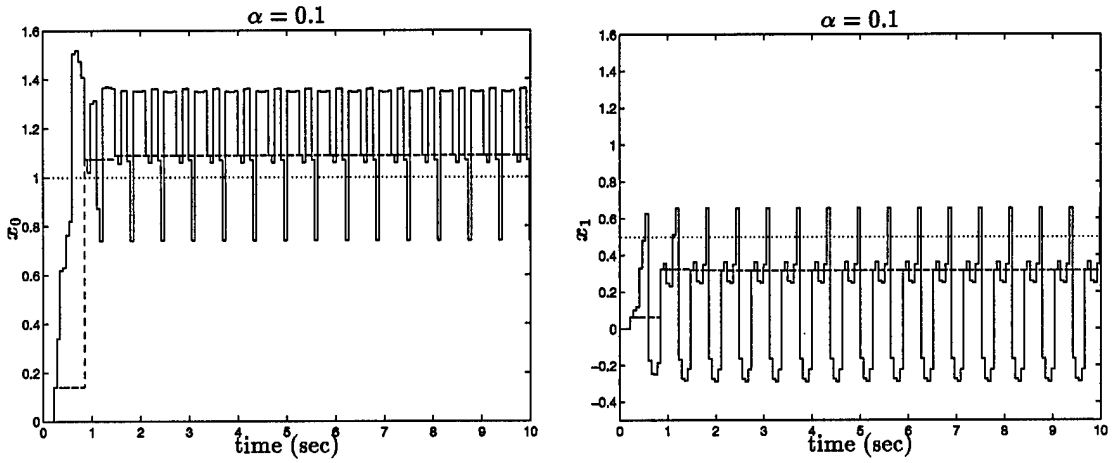


Figure 40 MCEU performance with baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of α, '—' represents MCEU estimates of α sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of α sampled once every cycle.

4.4.2.2 MCEU Performance for $\alpha = 0.5$. Figures 42 and 43 illustrate estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for $\alpha = 0.5$. The responses of the MCEU, for the different sampling rates, are identical to those described in the section where baseband mixing process was neglected (with the exception of the amplitudes of $\hat{x}_0$ and $\hat{x}_1$). Notice that the amplitudes of $\hat{x}_0$ and $\hat{x}_1$ are less than the true amplitudes of x_0 and x_1. This is a direct result of baseband mixing process. The decrease in estimated values are attributed to the low ratio of $f_o : 1/T_c$ in which harmonics of the higher frequency terms are not low pass filtered. These harmonics decrease the level of energy present in the correlator



a) MCEU estimation of x_0. b) MCEU estimation of x_1.

Figure 41 MCEU performance with baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of x_0 and x_1, '-' represents MCEU estimates of x_0 and x_1 sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of x_0 and x_1 sampled once every cycle.

samples resulting in estimates of $\hat{x}_0 < x_0$ and $\hat{x}_1 < x_1$. Since the correlator measurements of each arm, maintain a fixed ratio with respect to each other, $\hat{\alpha}$ is not degraded. Smaller estimates of $\hat{x}_0$ and $\hat{x}_1$, relative to true parameters, occurred for all cases of α tested; the results of these test cases are summarized in Section 4.4.3.

It should be noted that underestimating the values of $\hat{x}_0$ and $\hat{x}_1$ should not present a significant problem in the code tracking performance of the MCTL since the same baseband conversion process applies to the MCTL. As such, the relative amplitudes of the signals processed in the MCTL will be proportional to the amplitudes of the signals processed in the MCEU.

4.4.2.3 MCEU Performance for $\alpha = 1.3$. Figures 44 and 45 provide very similar results to those previously described. A significant difference is the estimated value $\hat{\alpha}$, illustrated in Figure 44, which has larger estimation error relative to the estimates $\hat{\alpha} \in [.2, 1.0]$. This increased estimation error is present for $\hat{\alpha} \in (1.0, 1.5]$. Note that at the sample rate of $1/.63\ Hz$, estimates are collected from the lower bound of the response curve envelope. The true value of α is located at the center of the MCEU response curve

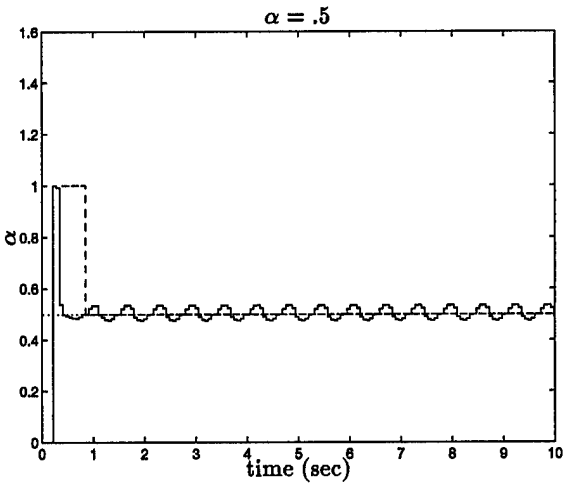
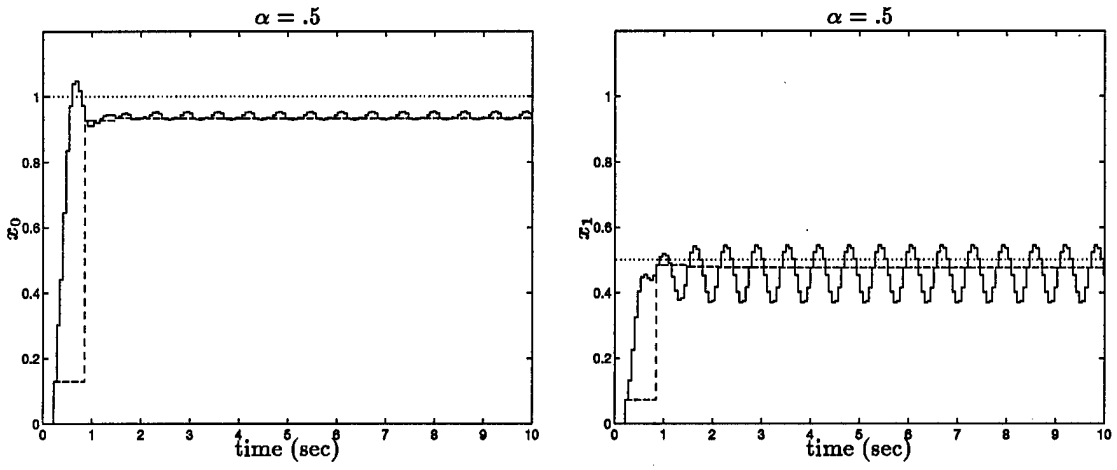


Figure 42 MCEU performance with baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of α, '—' represents MCEU estimates of α sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of α sampled once every cycle.

envelope. For this case, the epochs at which samples were taken led to a less accurate $\hat{\alpha}$, relative to the $\alpha = 0.5$ case.

4.4.3 Summary of MCEU Performance . The MCEU was tested for values of α at increments of 0.1 through the range $\alpha \in [0.1, 1.5]$, with and without the baseband mixing operation (BBMO) implemented. Based on the results summarized in Table 6, Figure 46 provides the estimates of α and Figure 47 the estimates of x_0 and x_1, each versus the actual values. Figure 46 illustrates the accuracy potential of the MCEU. All estimates with the exception the estimate of $\alpha = 0.1$ are within 0.03 chips of the true value of α. The figure also illustrates, as previously discussed, that the baseband conversion process does not affect the estimates of α. Figure 47, however, indicates that the baseband conversion process does affect the estimates of x_0 and x_1. Furthermore, with the exception of the estimate for $\alpha = 0.1$, the difference between the estimates of $\hat{x}_0$ and $\hat{x}_1$ with baseband conversion versus no baseband conversion is nearly constant throughout the range of α tested. In summary, $\hat{\alpha}$ is determined by the relative differences between the values of R_k; whereas, $\hat{x}_0$ and $\hat{x}_1$ are directly affected by the amplitudes/energy levels at the correlator inputs.



a) MCEU estimation of x_0.

b) MCEU estimation of x_1.

Figure 43 MCEU performance with baseband conversion mixing process (assuming perfect tracking) where '-.' represents the actual value of x_0 and x_1, '—' represents MCEU estimates of x_0 and x_1 sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of x_0 and x_1 sampled once every cycle.

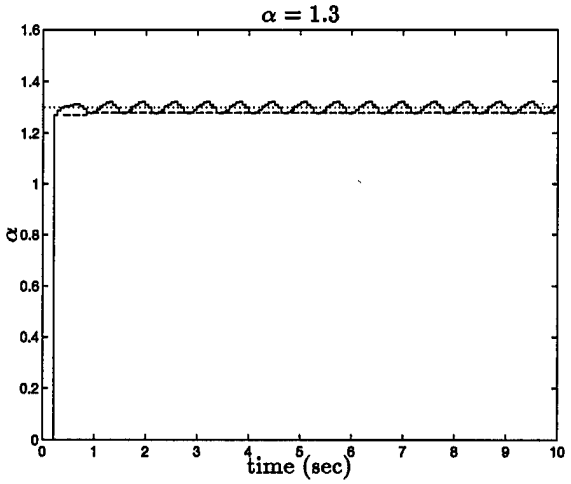
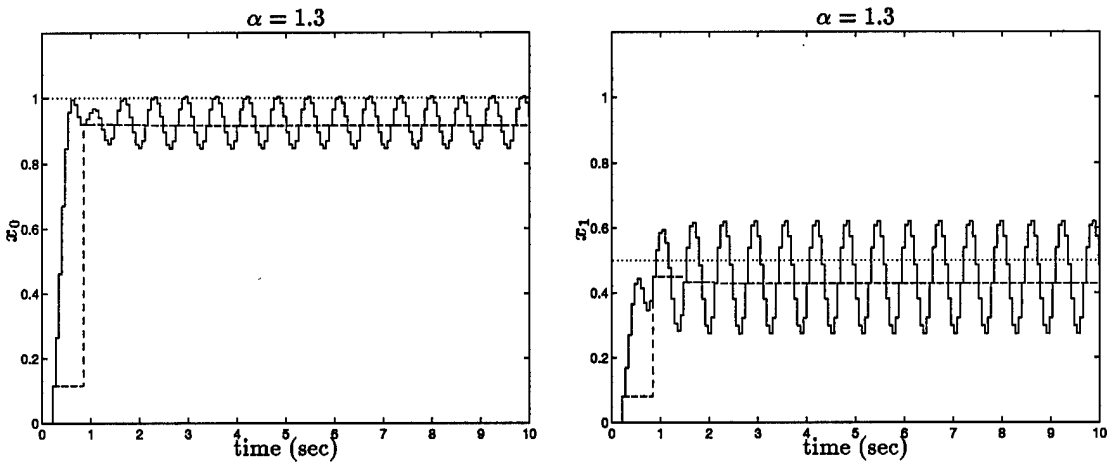


Figure 44 MCEU performance with baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of α, '—' represents MCEU estimates of α sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of α sampled once every cycle.



a) MCEU estimation of x_0.

b) MCEU estimation of x_1.

Figure 45 MCEU performance with baseband conversion mixing process (assuming perfect tracking) where '.' represents the actual value of x_0 and x_1, '—' represents MCEU estimates of x_0 and x_1 sampled every one-tenth of a cycle, and '- -'represents MCEU estimates of x_0 and x_1 sampled once every cycle.

Table 6 Summary of MCEU performance where α, $x_0 = 1.0$, and $x_1 = 0.5$ are the true parameters.

	with BBMO			without BBMO		
α	$\hat{\alpha}$	$\hat{x}_0$	$\hat{x}_1$	$\hat{\alpha}$	$\hat{x}_0$	$\hat{x}_1$
0.1000	0.1500	1.0883	0.3161	0.1500	1.1633	0.3380
0.2000	0.1974	0.9332	0.4701	0.1974	0.9976	0.5026
0.3000	0.3000	0.9346	0.4708	0.3000	0.9990	0.5033
0.4000	0.4013	0.9362	0.4699	0.4013	1.0007	0.5024
0.5000	0.5000	0.9328	0.4762	0.5000	0.9971	0.5091
0.6000	0.5926	0.9300	0.4766	0.5926	0.9941	0.5096
0.7000	0.6997	0.9360	0.4709	0.6997	1.0005	0.5034
0.8000	0.7995	0.9361	0.4704	0.7995	1.0007	0.5029
0.9000	0.8994	0.9361	0.4704	0.8994	1.0007	0.5029
1.0000	0.9912	0.9323	0.4659	0.9912	0.9967	0.4981
1.1000	1.0758	0.9296	0.4507	1.0758	0.9937	0.4819
1.2000	1.1881	0.9228	0.4476	1.1881	0.9865	0.4786
1.3000	1.2785	0.9160	0.4286	1.2785	0.9792	0.4583
1.4000	1.3736	0.9092	0.4078	1.3736	0.9720	0.4361
1.5000	1.4892	0.9024	0.4061	1.4891	0.9647	0.4342

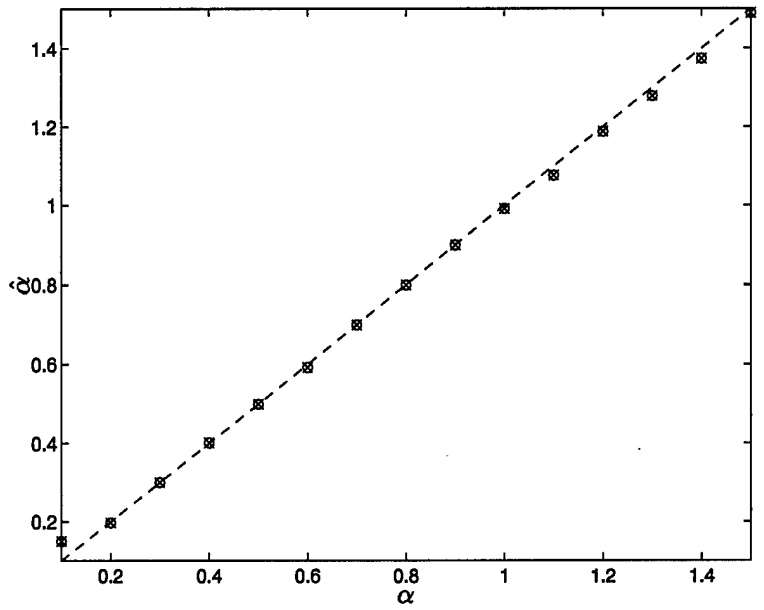


Figure 46 MCEU estimates of $\hat{\alpha}$ versus actual values of α (MCEU sampling period: 0.63 *sec*). Note: 'o' represents MCEU estimates with mixing operation; 'x' represents MCEU estimates without mixing operation. (MCEU sampling period: 0.63 *sec*).

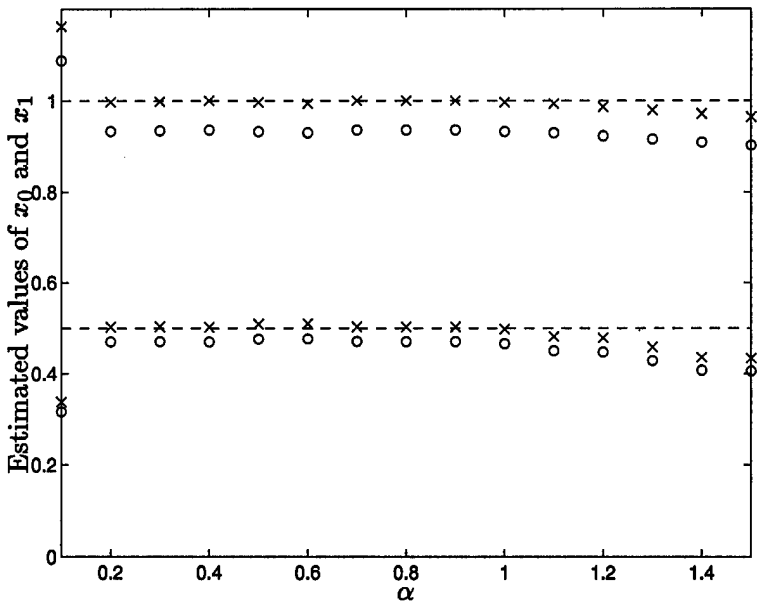
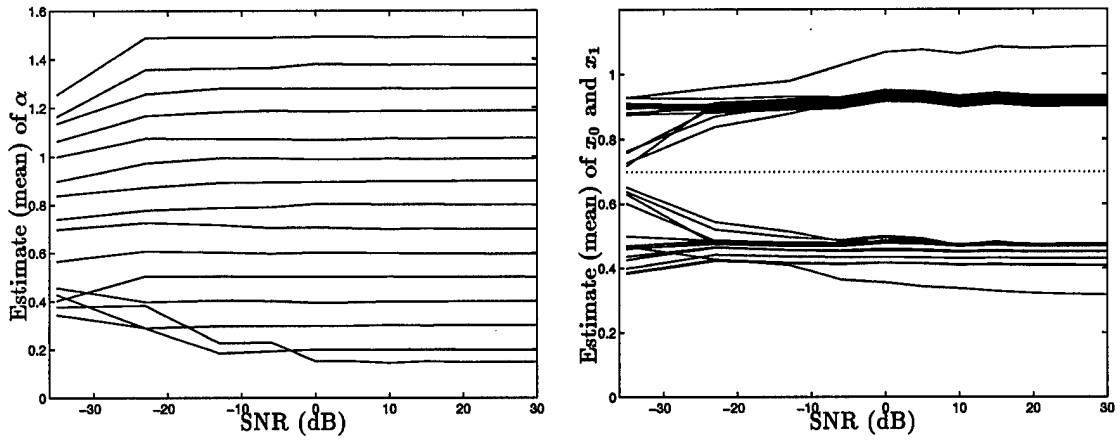


Figure 47 Estimated values of x_0 and x_1. Note: 'o' represents MCEU estimates with mixing operation; 'x' represents MCEU estimates without mixing operation. (MCEU sampling period: 0.63 *sec*).

4.5 Simulation # 2: MCEU Performance in AWGN

Since the performance baseline of the MCEU has been characterized under ideal conditions, the MCEU performance will now be examined in the presence of AWGN. In particular, it is desired to determine the SNR threshold at which the MCEU performance begins to significantly degrade. SNR levels were varied through the range of SNR$\in [-35, 30]$ dB as specified in Table 5. Once again, the MCTL is assumed perfectly synchronized with the received signal. MCEU samples are output every 0.63 *sec* over a period of 10 *sec*, providing 15 estimates for analysis.

Figure 48 illustrates the mean of the estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for delays of $\alpha \in [0.1, 1.5]$ in increments of 0.1 (NOTE: These figures are only presented to show trends, not to provide detail). Each line of Figure 48a represents the mean of the estimates of an $\alpha \in [0.1, 1, 5]$. The figure indicates that the mean of $\hat{\alpha}$ and $\hat{x}_0$, $\hat{x}_1$ are approximately constant for SNR$\in [-23, 30]$ dB with the exception of the estimates of delays $\alpha = 0.1$ and $\alpha = 0.2$. The mean of the estimates at these delays show degradation at SNR≤ 0 dB and SNR≤ -13 dB, respectively.

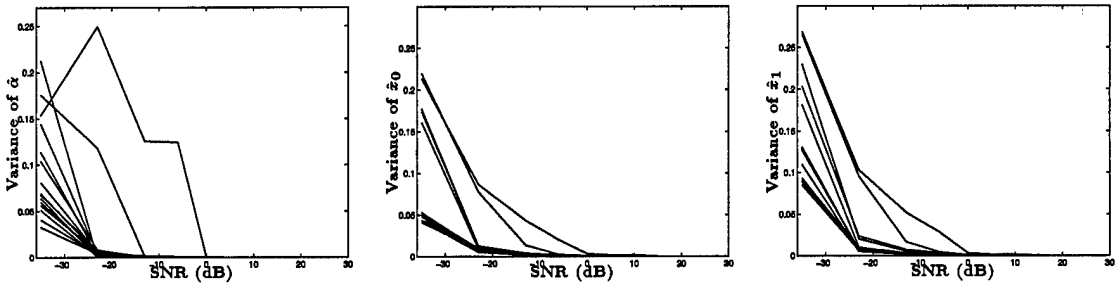


a) Mean of $\hat{\alpha}$. b) Mean of $\hat{x}_0$ and $\hat{x}_1$.

Figure 48 Mean of estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ provided by MCEU for delays $\alpha \in [0.1, 1.5]$.

Figure 49 illustrates the variance of the estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for delays of $\alpha \in [0.1, 1.5]$ in increments of 0.1 (NOTE: These figures are only presented to show trends, not to provide detail). Variance degradation is significant for SNR≤ -23 dB for all α.

Notice that the change in the slopes of the variance of the estimates for SNR≤ -23 dB differ considerably through the range of delays. The variance degradation for $\alpha = 0.1$ and $\alpha = 0.2$ begins at 0 dB and -13 dB, respectively.



a) Variance of $\hat{\alpha}$. b) Variance of $\hat{x}_0$. c) Variance of $\hat{x}_1$.

Figure 49 Variance of estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ provided by MCEU for delays $\alpha \in [0.1, 1.5]$.

In order to more closely examine the noise threshold effect, five delays ($\alpha = 0.3$, $\alpha = 0.5$, $\alpha = 0.7$, $\alpha = 1.0$, and $\alpha = 1.3$) were examined in detail through the full range of SNRs. These delays represent a uniform sampling of the 15 delays previously considered. For these delays, the root mean square error (rmse) of 15 samples taken over a 10 second window is used as the measure of performance. The rmse reflects both the mean error as well as the variance of the estimates.

Figures 50 and 51 illustrate the rmse of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for the five delays examined. As with the mean and the variance previously discussed, significant degradation in performance occurs for SNR≤ -23 dB. In addition, note that little or no noise effects are observed when SNR$\geq$ 0 dB. The estimates $\hat{\alpha}$ for $\alpha = 1.3$ are most affected by increased SNR levels; however, this is also true for the noise-free case presented in Simulation # 1.

Figures 52 and 53 illustrate the rmse of estimates for all delays $\alpha \in [0.1, 1, 5]$ incremented by 0.1 at SNR levels of -23 dB, 0 dB, and 10 dB. These figures illustrate that there is a much larger MCEU performance degradation between 0 dB and -23 dB than between 10 dB and 0 dB. This is consistent with the previously determined noise threshold of SNR=-23 dB.

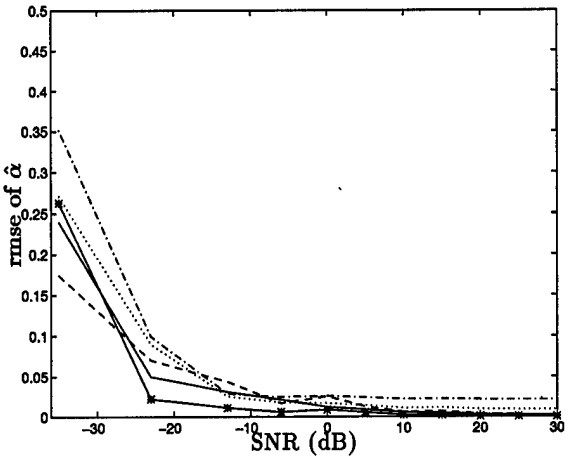


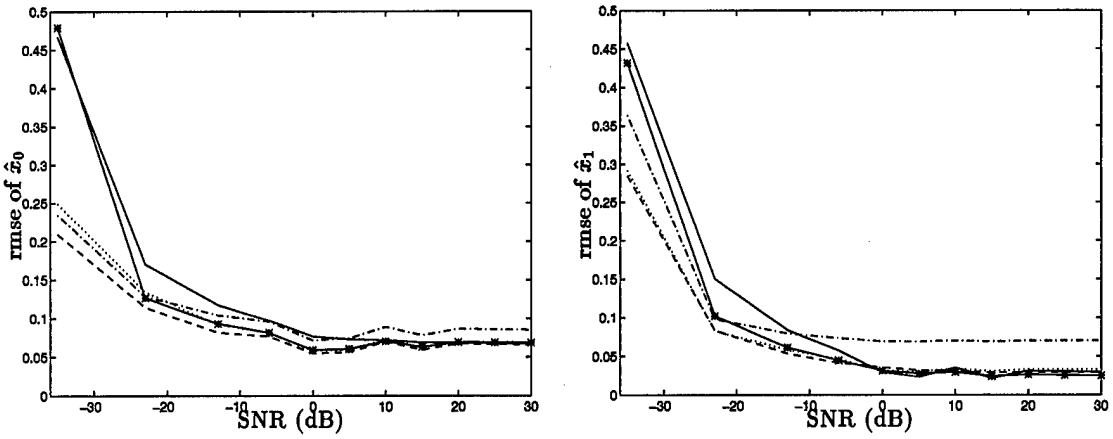
Figure 50 rmse of $\hat{\alpha}$ provided by the MCEU for $\alpha = 0.3$ ('—'), $\alpha = 0.5$ ('-*'), $\alpha = 0.7$ ('- -'), $\alpha = 1.0$ ('.'), and $\alpha = 1.3$ ('-.')

As previously indicated, MCEU performance is the most noise sensitive for the small reflected signal delays of $\alpha = 0.1$ and $\alpha = 0.2$; $\hat{\alpha}$ for delay $\alpha = 1.3$ shows the most sensitivity to increased noise levels. $\hat{\alpha}$ of $\alpha = 0.5$ is least affected by increased noise levels.

Other than the estimates of x_0 and x_1 for delays $\alpha = 0.1$ and $\alpha = 0.2$, $\hat{x}_0$ and $\hat{x}_1$ for delay $\alpha = 0.3$ show the largest rmse for estimates of x_0 and x_1. Degradation in the estimates $\hat{x}_0$ and $\hat{x}_1$ is uniform for a decrease in SNR from 10 dB to 0 dB; however, the increase in rmse due to a decrease in SNR from 0 dB to -23 dB is somewhat random for delays $\alpha \in [0.3, 1.5]$.

4.5.1 Summary of Results. When the MCTL is perfectly tracking the direct path code phase, the MCEU exhibits a rmse noise threshold of -23 dB, for reflected signal delays of $\alpha \in [0.3, 1.5]$; from Table 5, -23 dB corresponds to a computed SNR of -13 dB.

A comparison of the results from this analysis to GPS applications can be made by normalizing a typical GPS SNR. Spilker demonstrates that a typical GPS noise density level of $N_0 = 205.2\ dBW/Hz$ exists for an equivalent noise temperature of a typical low noise amplifier configuration [8]. Given that the bandwidth of the main lobe of the C/A spreading code is $B = 2.046\ MHz$, an approximation of the noise power at the input of a receiver antenna can be expressed as $N = N_0 B = -141.89\ dBW$. GPS requirements specify a minimum signal strength of $-160\ dBW$ for C/A code on the L1 carrier which



a) rmse of $\hat{x}_0$. b) rmse of $\hat{x}_1$.

Figure 51 rmse of $\hat{x}_0$ and $\hat{x}_1$ provided by the MCEU for $\alpha = 0.3$ ('—'), $\alpha = 0.5$ ('-*'), $\alpha = 0.7$ ('- -'), $\alpha = 1.0$ ('.'), and $\alpha = 1.3$ ('-.')

would provide an SNR$\approx -18dB$. From table 5, a specified SNR of -35 dB resulted in a calculated SNR of -19 dB. At this SNR, results from Simulation # 2 indicate significant degradation. Improvements in performance of the eMRDLL would most likely occur if the bandwidth of the LPF could reduce to a normalized value proportional value to the data rate $<< 1/NT_c$. In addition, using a code length N=1023 may provide improvement by increasing processing gain.

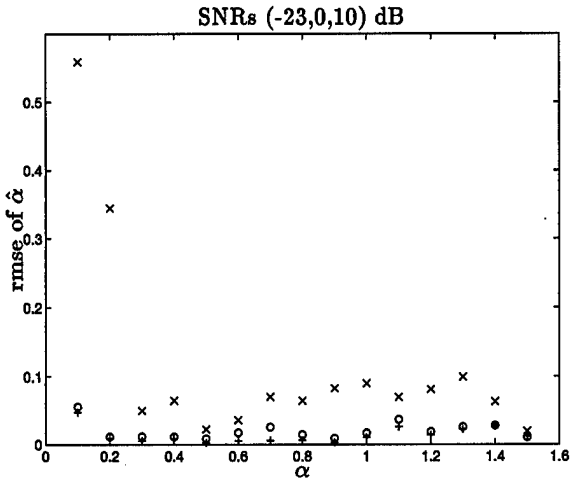
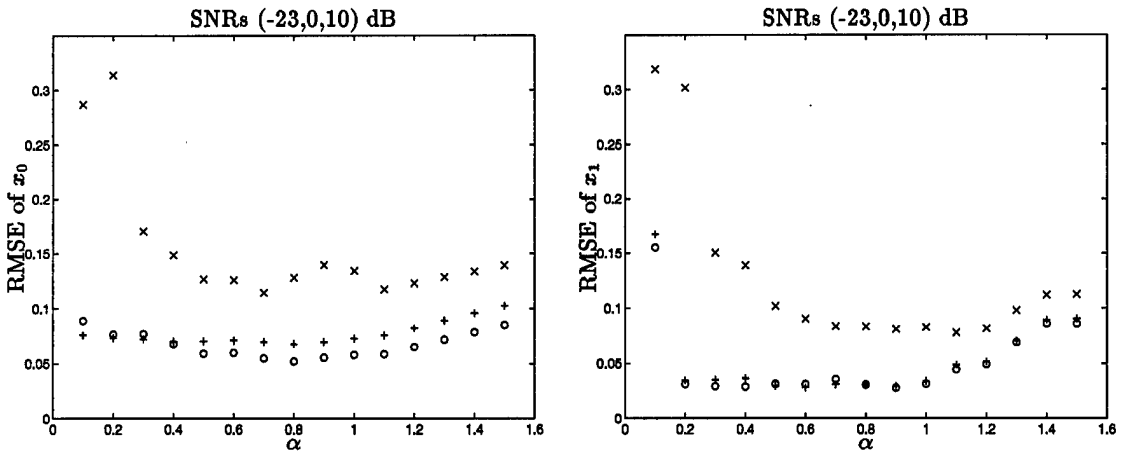


Figure 52 rmse of $\hat{\alpha}$ provided by the MCEU where 'x' represents -23dB, 'o' represents 0dB, and '+' represents 10dB.



a) rmse of $\hat{x}_0$. b) rmse of $\hat{x}_1$.

Figure 53 rmse of $\hat{x}_0$ and $\hat{x}_1$ provided by the MCEU where 'x' represents -23dB, 'o' represents 0dB, and '+' represents 10dB.

4.6 Simulation # 3: eMRDLL versus NCDLL (Noise-Free Comparison)

The objective of this section is to determine the noise-free performance of the eMRDLL in comparison with the NCDLL (normalized correlator spacings of $\Delta = 1.0$ and $\Delta = 0.1$). The normalized code phase error $(\tau_o(t) - \hat{\tau}_o(t))/T_c$ of the code generator with respect to the received signal is used as the measure of performance of each design. Simulations were conducted without AWGN in order to characterized the performance of each design under ideal conditions.

In order to isolate the performance of the MCTL, initial open loop tests were conducted assuming perfect synchronization by the VCC and perfect estimates from the MCEU. Tests were conducted with and without the baseband mixing operation implemented. Results indicate that a DC bias exists at the discriminator output, $e(\delta, t)$, as shown in Figure 54, for both cases. Although, the amplitudes of each of the errors signals in the figure are different, the bias present is approximately equal between both discriminator outputs. Therefore, the bias present is likely attributed to a Simulink computational anomaly in the evaluation of the correlation of the received direct path and reflected signals with the locally generated code.

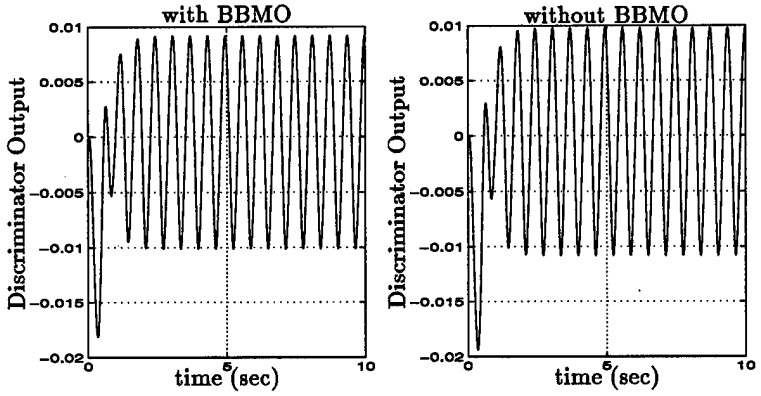
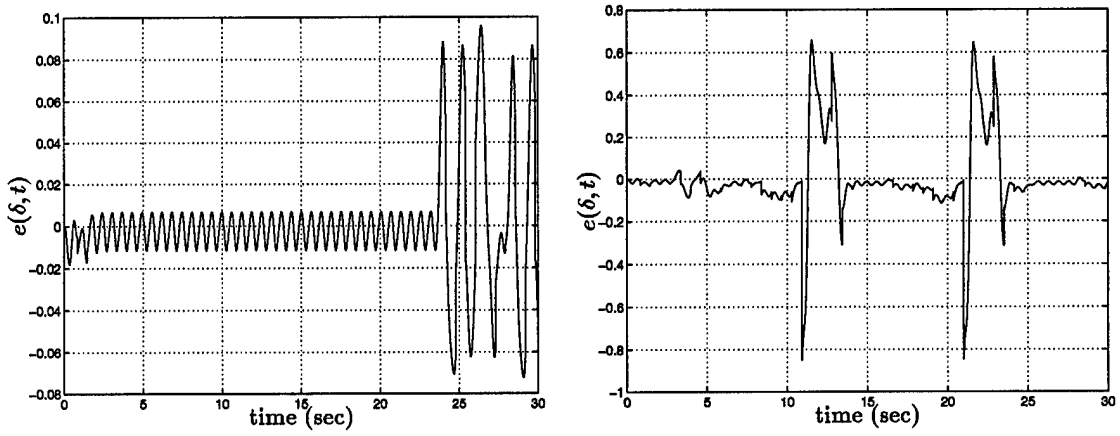


Figure 54 MCTL DC bias (with and without mixing process).

This bias has an adverse effect on the closed loop tracking performance of eMRDLL. The ALC operates similarly to a basic first order loop filter. Any DC component is integrated by the ALC, which drives the VCC away from the desired operating point. Because the bias is relatively small, a degradation in tracking performance does not occur

for the first several cycles in most of the test cases.[4] The bias has the least effect when the reflected signal delay is $\alpha = 0.3$, as shown in Figure 55a, in which the bias does not impact synchronization until $t \approx 23$ *sec* (≈ 36.5 code cycles). The bias has the greatest effect when the reflected signal delay is $\alpha = 0.5$, as shown in Figure 55b, in which the bias begins to significantly affect closed loop tracking synchronization after $t \approx 11$ *sec* (≈ 17.5 code cycles). For these reasons, analysis on the eMRDLL was conducted for the two time intervals of 0 to 10 *sec* and 0 to 30 *sec*; simulations were not conducted for longer periods of time in order minimize the simulation run times.

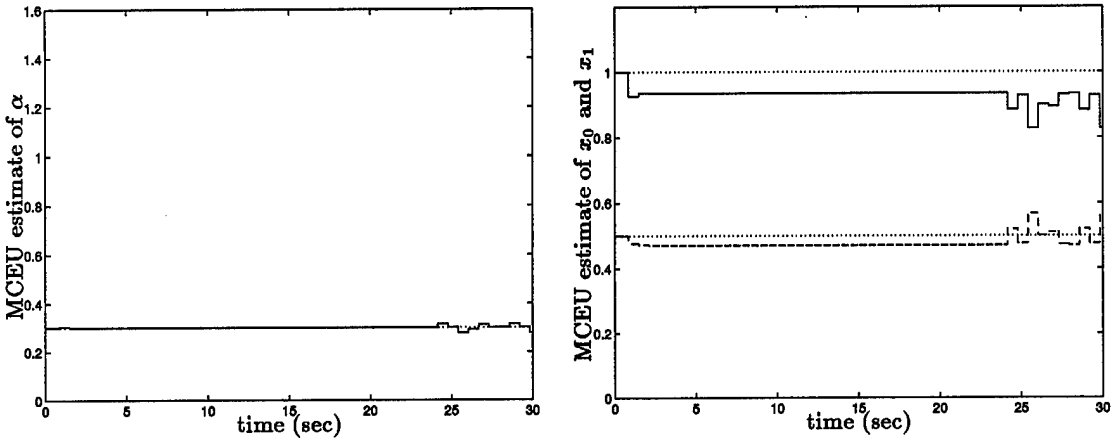


a) $\alpha = 0.3$ b) $\alpha = 0.5$

Figure 55 Discriminator outputs for $\alpha = 0.3$ and $\alpha = 0.5$.

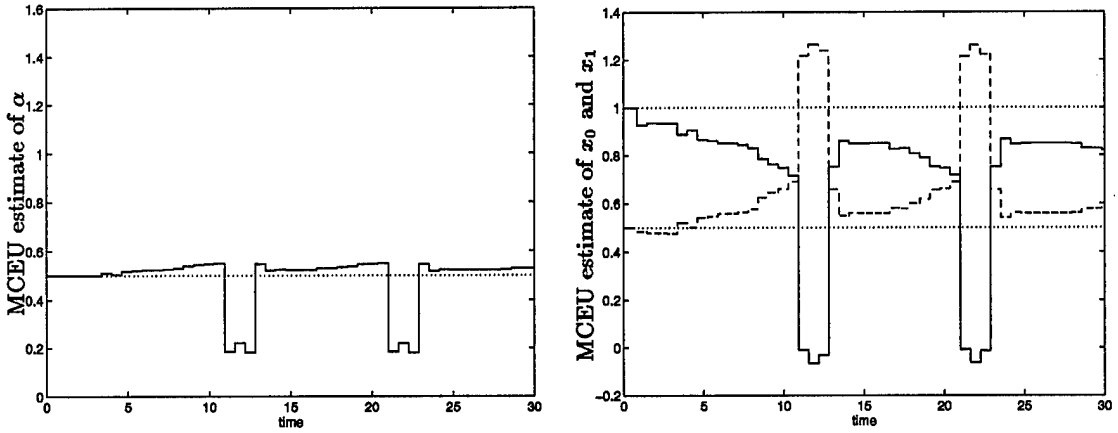
4.6.1 MCEU Performance During Code Phase Tracking. MCEU estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for $\alpha = 0.3$ and $\alpha = 0.5$ are illustrated in Figures 56 through 57. The figures indicate that the estimates degrade in proportion to the corresponding discriminator output; correspondingly, the degradation is less significant for $\alpha = 0.3$ and more severe for $\alpha = 0.5$. The eMRDLL is able to correct itself and maintain track as indicated by the cyclic nature depicted in Figure 55. This cyclic property is also present in Figure 57. Although a bias exists, the eMRDLL does not loose lock as a result of this bias.

[4]Because the Simulink models are discrete in nature, a certain VCC input threshold must be exceeded prior to any VCC frequency adjustment.



a) MCEU estimates of α. b) MCEU estimates of x_0 and x_1.

Figure 56 MCEU estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for $\alpha = 0.3$.



a) MCEU estimates of α. b) MCEU estimates of x_0 and x_1.

Figure 57 MCEU estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for $\alpha = 0.5$.

4.6.2 eMRDLL Versus NCDLL ($\Delta = 1.0$ and $\Delta = 0.1$) Noise-Free Code Phase Tracking. Code phase error tracking measurements were taken after the models reached steady-state. The simulations for the NCDLL were conducted for 50 *sec* to ensure that at least 10,000 samples were available for determining tracking performance. Shorter simulation run times were acceptable for the eMRDLL, because simulations were started with the eMRDLL initially synchronized.

Figure 58 provides the mean code phase tracking error of the NCDLL versus the eMRDLL. Because the operation of the MCEU is constrained to estimating multipath parameters for delays $\alpha \in [0.5, 1.5]$, the case in which only the LOS signal exists is not considered in the evaluation of the eMRDLL. The envelopes indicate the expected theoretical worst-case tracking error as determined by the equations in Table 1. The tracking performance of the NCDLL ($\Delta = 1.0$) is analyzed for uniformly spaced delays through the range of $\alpha \in [0, 1.5]$ in increments of 0.1; tracking performance was also analyzed for delays $\alpha = 0.55$ and $\alpha = 0.75$. A negative tracking error of the NCDLL corresponds to a reflected signal being out of phase with the LOS signal, as is the case here for $\alpha = 0.55$ and $\alpha = 0.75$. As expected, the NCDLL with correlator spacing $\Delta = 0.1$ outperforms the NCDLL with correlator spacing $\Delta = 1.0$; however, the results of the NCDLL with $\Delta = 0.1$ are higher ($\delta \approx 0.035$) through $\alpha \in [0.1, 1.0]$ than predicted by the equations in Table 1 ($\delta = 0.025$). Previous analysis has shown that a NCDLL with a normalized correlator spacing of $\Delta = 0.1$ does not outperform a spacing of $\Delta = 0.2$ in the presence of multipath [18]; perhaps the failure (of the NCDLL with $\Delta = 0.1$) to achieve predicted performance is a limitation intrinsic to the NCDLL. As illustrated in Figure 58, the mean code phase error of eMRDLL is lower than the NCDLL $\Delta = 0.1$ throughout the range of $\alpha \in [0.1, 1.0]$ with the exception of $\alpha = 0.5$ (for a final time $t_f = 10$ *sec*). For $t_f = 30$ *sec*, the code phase error for α =0.5, 0.6, and 0.7 is worse than results obtained for the NCDLL ($\Delta = 0.1$), indicating that, for longer periods of code phase tracking, the accumulating effect of the bias leads to greater degradation in eMRDLL tracking performance.

The code phase tracking rmse[5], commonly known as tracking jitter, for the eMRDLL and the NCDLL is illustrated in Figure 59. The code phase tracking rmse for the NCDLL

[5]The rmse reflects the effects of the bias in addition to the variance of the code phase tracking error.

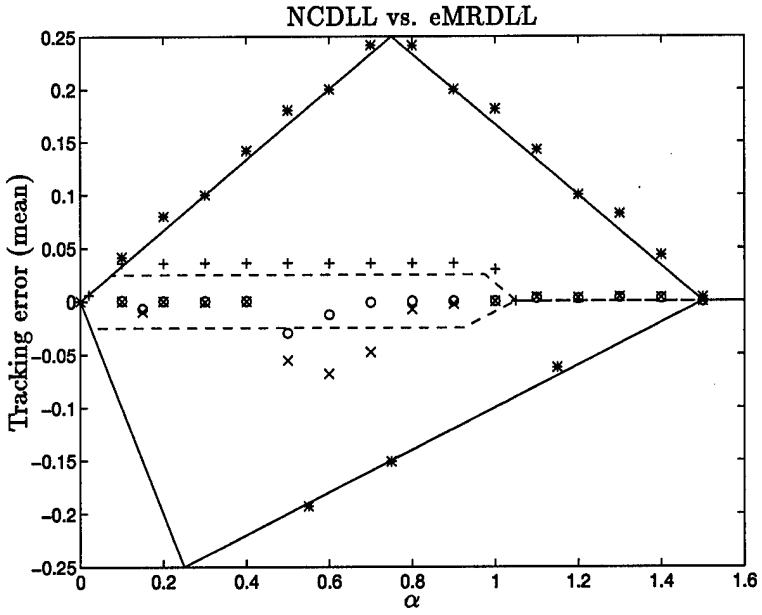


Figure 58 Code phase tracking error (mean) where '*' represents NCDLL ($\Delta = 1.0$), '+' represents NCDLL ($\Delta = 0.1$), 'o' represents eMRDLL (0 to 10 sec), 'x' represents eMRDLL (0 to 30 sec).

($\Delta = 1.0$ and $\Delta = 0.1$) is approximately the same as the absolute value of the mean phase error which indicates that negligible bias is present in their measurements. The rmse phase error of the eMRDLL is greater than the mean phase error, as a result of the bias. Results indicate that for $t_f = 10$ *sec*, the eMRDLL outperforms the NCDLL ($\Delta = 0.1$) for $\alpha \in [0.1 : 1.0]$ with the exception of $\alpha = 0.5$. As with the mean phase error, the eMRDLL performance degrades when $t_f = 0 : 30$ *sec* for $\alpha = 0.5$, 0.6, and 0.7 where it is outperformed by the NCDLL ($\Delta = 0.1$). The code phase tracking rmse for other values of α increase only slightly due to the t_f increase. It should be noted that the NCDLL ($\Delta = 0.1$) marginally outperforms eMRDLL for $\alpha \in [1.1, 1.4]$; this is a reflection of the sampling error of the MCEU as demonstrated in Simulation # 1 for $\alpha = 1.3$ (in which samples were taken from the lower bound of the MCEU output). This sampling error is present for all $\alpha \in [1.1, 1.4]$. The eMRDLL could possibly be improved by reducing its normalized correlator spacing Δ. Overall, these noise-free baseline results indicate that the eMRDLL performed much better than the NCDLL ($\Delta = 1.0$) for all $\alpha \in (0, 1.5]$. One may speculate that if Simulink computational error is solely responsible for the observed

bias, then the eMRDLL could, uniformly in α, outperform the NCDLL ($\Delta = 0.1$) in an actual receiver.

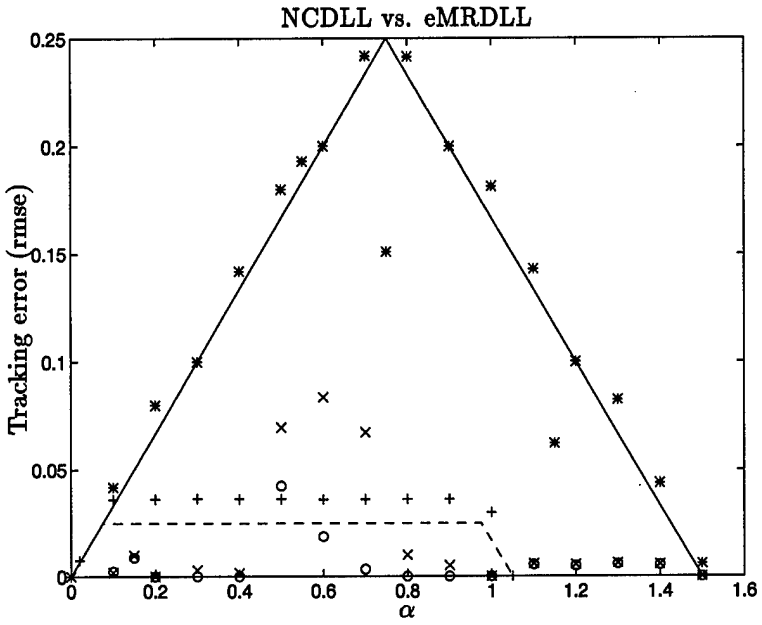


Figure 59 Code phase tracking rmse where '*' represents NCDLL, $\Delta = 1.0$. '+' represents NCDLL, $\Delta = 0.1$. 'o' represents eMRDLL (0 to 10 sec), 'x' represents eMRDLL (0 to 30 sec).

4.7 Simulation # 4: eMRDLL Performance in AWGN

Simulations were conducted to characterize the performance of the eMRDLL in the presence of AWGN. The objective was to determine the effects of different SNR levels on the eMRDLL code phase tracking. Five reflected signal delays, $\alpha = 0.3$, 0.5, 0.7, 1.0, and 1.3, (the same delays analyzed in Simulation # 2) were used for simulation. The values of α chosen for analysis represent the best ($\alpha = 0.3$) through the worst ($\alpha = 0.5$) code phase tracking results of Simulation # 3. As before, the range $t_f := 10$ *sec* was used for analysis; however, the $t_f := 20$ *sec* was used instead of $t_f := 30$ *sec*, to decrease simulation run times. The simulations were conducted for a sampling of SNR$\in [-35, 30]$ dB, as indicated in Table 5.

The mean of the code phase tracking error for both ranges of t is provided in Figure 60. The figure illustrates the distinct differences in the fluctuation of the mean errors

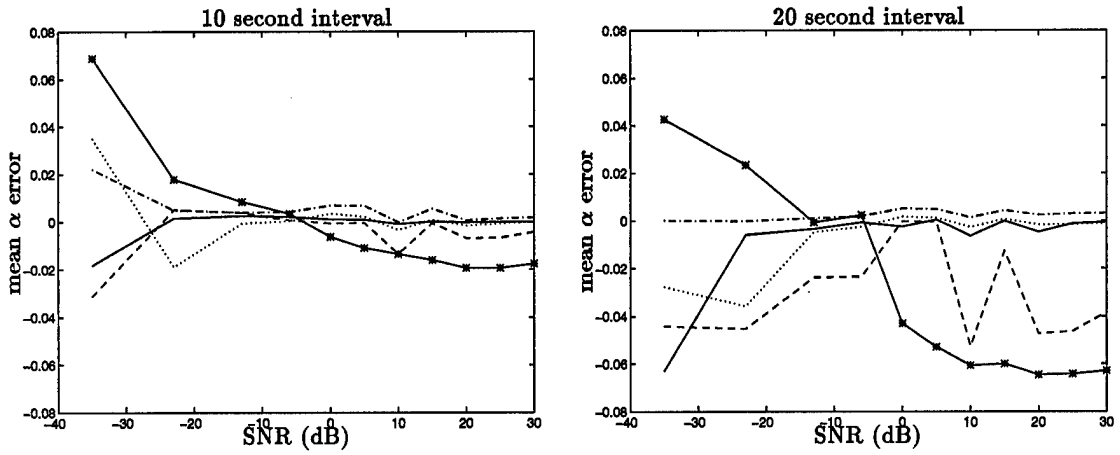
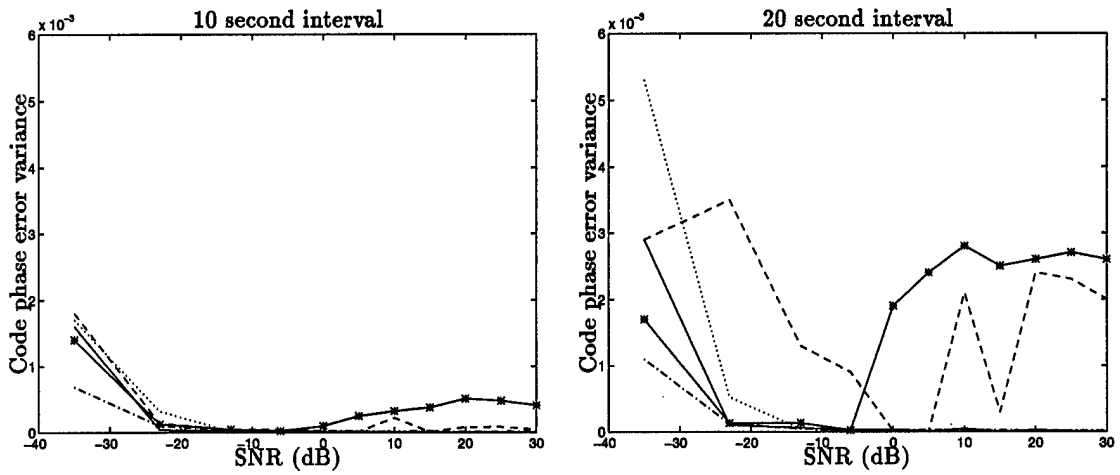


Figure 60 Mean of code phase tracking error for $\alpha = 0.3$ ('—'), $\alpha = 0.5$ ('-*'), $\alpha = 0.7$ ('- -'), $\alpha = 1.0$ ('.'), and $\alpha = 1.3$ ('-.').

over the different intervals of time. For $t_f := 10$ *sec*, the mean error is relatively constant for all delays through the range SNR$\in [-13, 30]$ *dB*, with the exception of $\alpha = 0.5$ (which was shown to be affected by the bias after five seconds; see Figure 55b). At SNR=-13 *dB*, the mean code phase tracking error for $\alpha = 1.0$ begins to grow as a result of the noise. At SNR=-35 *dB*, the mean tracking error at all of the delays has been affected by the noise. For $t_f := 20$ *sec*, the mean tracking error versus SNR fluctuates more dramatically for $\alpha = 0.5$ and $\alpha = 0.7$; this is consistent with Figure 59, wherein $\alpha = 0.7$ was shown to have a significantly larger rmse for longer simulation runs. The mean tracking error of the remaining delays are relatively constant throughout the range SNR$\in [-13, 30]$ *dB*. Note that for $\alpha = 0.3$ the tracking error did not show degradation until SNR=-23 *dB*. For $\alpha = 0.5$ and $\alpha = 0.7$, SNR$\in [-13, -6]$ *dB* and SNR$\in [0, 5]$ *dB*, respectively, the mean tracking error was minimal; this illustrates how the noise exciting the system can actually mitigate the bias effect.[6]

Similar results were observed for the variance of the code phase tracking error as illustrated in Figure 61. For $t_f := 10$ *sec*, Figure 61a, the variance clearly exhibits a SNR threshold of -23 *dB*. For $t := 20$ *sec*, this is also true with the exception of $\alpha = 0.7$ which shows an erratic variance versus SNR. As with the mean error, the variance of the

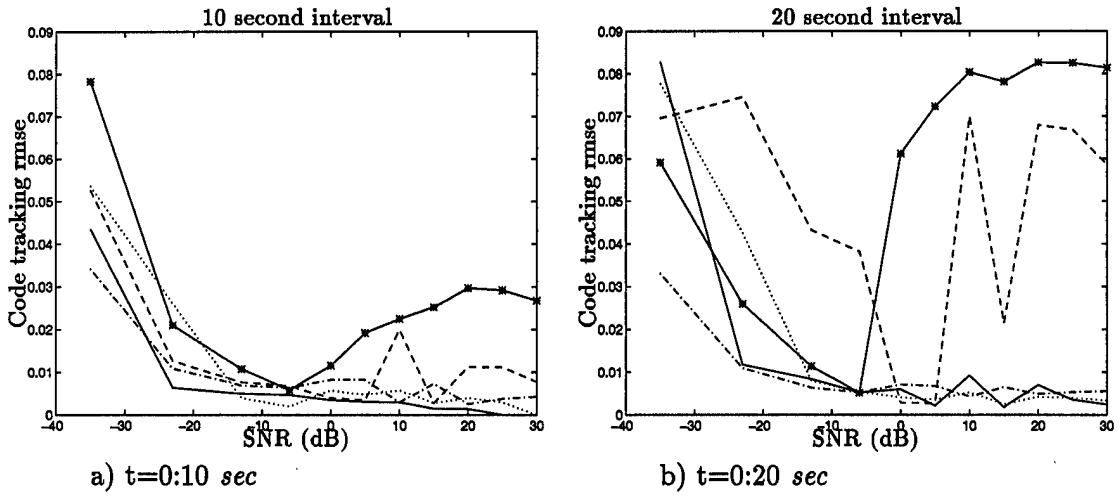
[6]This is known as a 'persistence of excitation' condition.



a) t=0:10 *sec* b) t=0:20 *sec*

Figure 61 Variance of code phase error for $\alpha = 0.3$ ('—'), $\alpha = 0.5$ ('-*'), $\alpha = 0.7$ ('- -'), $\alpha = 1.0$ ('.'), and $\alpha = 1.3$ ('-.').

code phase tracking error for $\alpha = 0.5$ and $\alpha = 0.7$ was actually reduced in the range of SNR$\in [0, 30]$ dB; the presence of noise counteracts the detrimental effects of the bias in this range.



a) t=0:10 *sec* b) t=0:20 *sec*

Figure 62 Code tracking rmse for $\alpha = 0.3$ ('—'), $\alpha = 0.5$ ('-*'), $\alpha = 0.7$ ('- -'), $\alpha = 1.0$ ('.'), and $\alpha = 1.3$ ('-.').

An alternative measure of code tracking performance is the code phase tracking rmse or tracking jitter, which reflects the bias and variance of the error signal. Figure 62 provides the code phase rmse for $t_f := 10$ *sec* and $t_f := 20$ *sec*. For $t_f := 10$ *sec*, the rmse of the

code phase tracking error exhibits a threshold at SNR≈ -23 dB. For $t_f := 20$ sec, the SNR performance threshold drops to -13 dB with the exception of $\alpha = 0.5$ and $\alpha = 0.7$ which display erratic rmse behavior versus SNR.

In summary, the addition of moderate levels of noise can actually improve the performance of the eMRDLL. The noise has the effect of perturbing or exciting the system such that the effects of the Simulink bias are reduced. Of course, there is a noise threshold for which the code phase tracking of eMRDLL begins to degrade. From the figures in this section, one can see that SNR level at which the performance of the eMRDLL degrades varies for different delays; however, generally an SNR of -23 dB (relative to the Simulink models) is the approximate SNR level at which significantly degradaded code tracking performance can be expected for $t_f := 10$ sec. An SNR threshold of -13 dB (relative to the Simulink models) is expected when the interval for averaging is doubled.

As explained in the summary of Simulation # 2, the specified SNR level that would be equivalent to GPS SNR levels is ≈ -35 dB. Results show that at this SNR, significant degredation in tracking performance occurs. Reducing the bandwidth of the LPFs and increasing the length of the PN sequence to N=1023 may improve performance.

4.8 Simulation # 5: Time Varying Reflections

Previous simulations were conducted under the assumption that the delay of the reflected signal was slowly time varying, as would be encountered with static position determinations. Under this condition, it was assumed that the multipath delay over relatively short periods of time was constant.

Many realistic conditions involve time varying multipath delays. Simulations were conducted on the MCEU given a time varying multipath environment, wherein, it was assumed that the MCTL maintained perfect synchronization with the received signal. In addition, it was assumed that the received signal has already been perfectly filtered to baseband (i.e., carrier modulation was removed).

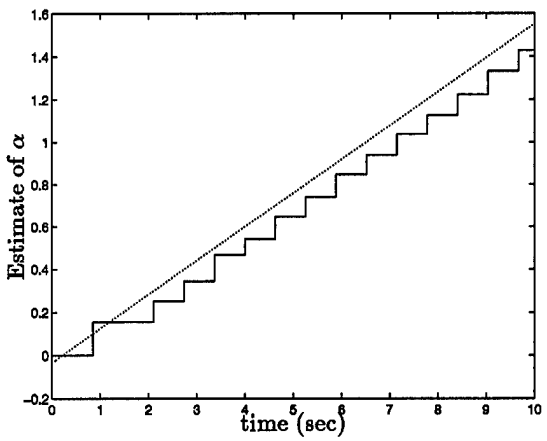
Two time varying multipath scenarios are considered.

- Case 1: $\alpha(t) = (10/63)t - 2.17/63$ for $t = 0 : 10$ *sec*
- Case 2: $\alpha(t) = (5/63)t - 1.085/63$ for $t = 0 : 19.5$ *sec*

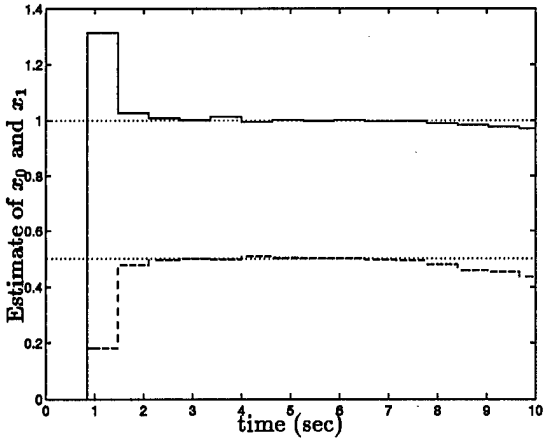
Time duration is chosen to ensure that delays of $\alpha \in [0, 1.5]$ are encountered within the simulation time. Estimates are calculated by the MCEU every 0.63 *sec.*

Figure 63 illustrate the results of the simulations conducted for case 1. Figure 63a indicates that the estimate of α is lower than the true for all α, with the exception of $\alpha = 0.1$. This under estimation of α is likely a result of the narrow bandwidth of the LPFs in the bank of correlators, because the narrow bandwidth LPFs invoke a time delay on the filtered signal; consequently, current estimates are based upon previous correlator samples which have been delayed by the lag/response of the LPF. The figure also indicates that $\hat{x}_0$ and $\hat{x}_1$ are very accurate with the exception of the estimates taken during the first two cycles, which is the transient period. Note that the increased error in the estimates $\hat{x}_0$ and $\hat{x}_1$ for $\alpha \in [1.1, 1.5]$ is consistent with results presented in Simulation # 1.

Because of the lag in the MCEU estimates observed in the first case, a slower time varying simulation was chosen for case 2, which has a delay which increases at half the rate of the first case. As illustrated in Figure 64, with the exception of the first estimate, the estimates of α are indeed closer to the true delays, as expected, since the change in α over time is less rapid. Note that the estimates $\hat{x}_0$ and $\hat{x}_1$ are nearly identical to those of



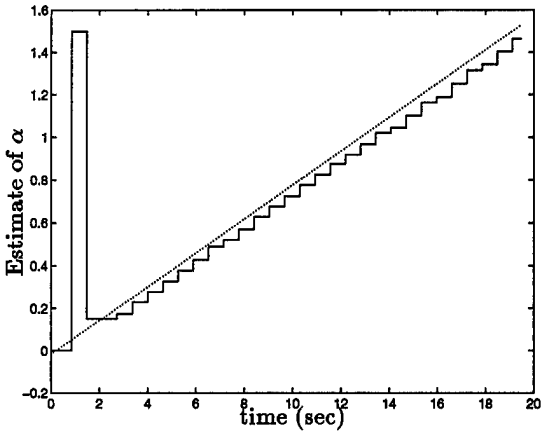
a) Estimates of α.



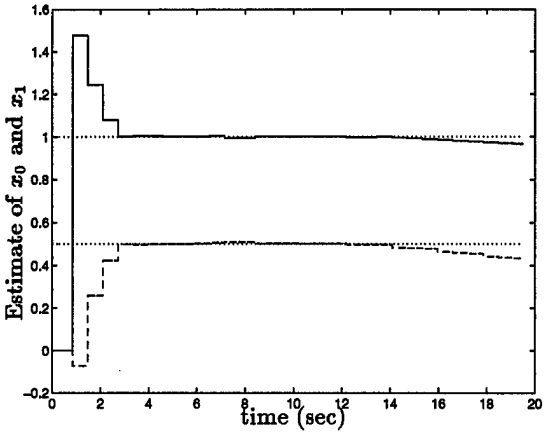
b) Estimates of $x_0 = 1.0$ ('—') and $x_1 = 0.5$ ('- -').

Figure 63 Time varying multipath estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$.

the first case, which suggests that the estimates in the range of $\alpha \in [1.1, 1.5]$ are degraded due to the instantaneous value of $\hat{alpha}$ as opposed to any time varying properties of the reflected signal.



a) Estimates of α.

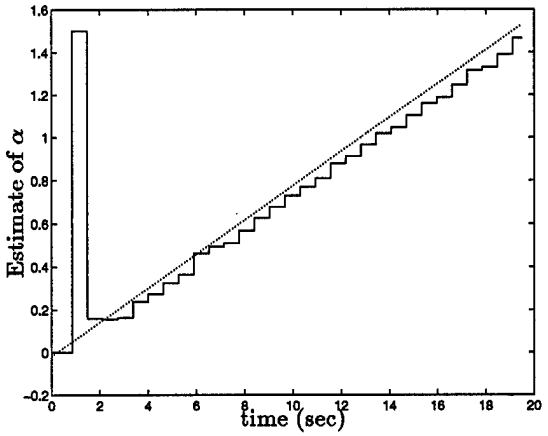


b) Estimates of $x_0 = 1.0$ ('—') and $x_1 = 0.5$ ('- -').

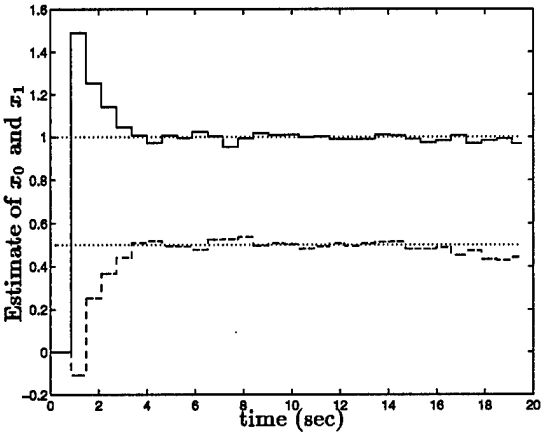
Figure 64 Time varying multipath estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$.

The effects of noise upon time varying multipath estimation were also considered. The SNR levels considered, 0 dB and -23 dB, were selected because previous results indicate that slight noise degradation occurs at 0 dB and significant degradation occurs at -23 dB (the SNR threshold). Figure 65 illustrates that, as expected, a 0 dB SNR has

little effect on the estimates of α. However, there is a slight degradation in the estimates $\hat{x}_0$ and $\hat{x}_1$.



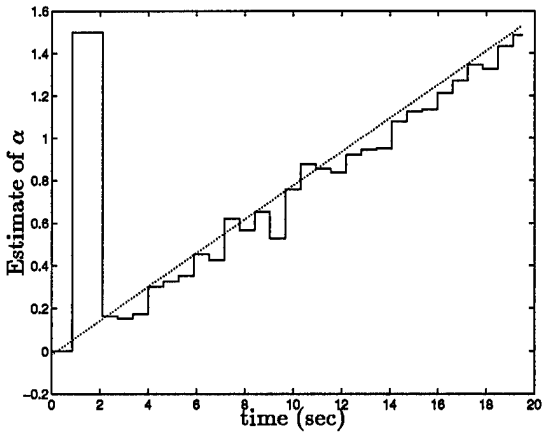
a) Estimates of α.



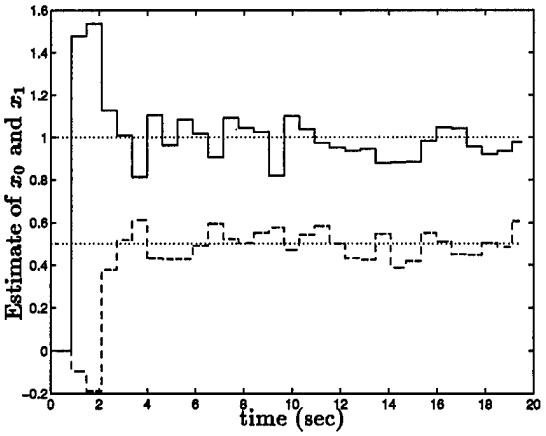
b) Estimates of $x_0 = 1.0$ ('—') and $x_1 = 0.5$ ('- -').

Figure 65 Time varying multipath estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for SNR=0 *dB*.

Figure 66 illustrates that decreasing the SNR level to -23 *dB* does begin to significantly effect the estimates of $\hat{\alpha}$, $\hat{x}_0$ and $\hat{x}_1$, as expected. The overall results indicate that the performance of the MCEU with time varying multipath is consistent with the performance encountered under a constant multipath delay, both with and without the presence of AWGN.



a) Estimates of α.



b) Estimates of $x_0 = 1.0$ ('—') and $x_1 = 0.5$ ('- -').

Figure 66 Time varying multipath estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ for SNR=0 *dB*.

V. Conclusion and Recommendations

5.1 Overview

This thesis proposed the use of maximum likelihood estimation in a code tracking loop design to mitigate the effects of multipath. Although the NCDLL with a normalized narrow correlator spacing of $\Delta = 0.1$ has been showed to reduce code phase tracking error caused by multipath relative to a standard ($\Delta = 0.1$) NCDLL, it cannot fully remove the tracking error. The eMRDLL, which is an enhanced version of the MRDLL [6,7], accounts for the parameters of the reflected as well as the direct path signals to mitigate the effects of multipath. The effectiveness of the eMRDLL is highly dependent on the performance of the MCEU, which is a MVU estimator capable of reaching the CRLB.

Analysis conducted on the equations which describe the operation of a standard NCDLL characterized a tracking curve known as the S-curve. Given a direct path only signal, the steady state tracking point of this S-curve is located at $\delta^* = 0$; however, in a multipath environment, the S-curve becomes distorted and tracks with a bias $\delta^* \neq 0$. Further analysis confirmed that reducing the correlator spacing from $\Delta = 1.0$ to $\Delta = 0.1$ reduced the effects of multipath significantly; however, a threshold was reached for which tracking errors could not be reduced further due to inherent constraints of the NCDLL design.

The MRDLL, which is composed of the MCTL, MCEU, and the ALC, has previously been shown to mitigate the effects of multipath [6,7]; however, operation of the MRDLL was limited to estimating multipath delays through the range of $\alpha \in [0.1, 1, 5]$ in increments of 0.1. The eMRDLL provides enhancements to the MRDLL, via an enhanced MCEU, which permits multipath parameters to be estimated through the full range of $\alpha \in [0.1, 1.5]$. An analysis was conducted to examine the performance improvements of the eMRDLL in a multipath environment. Analysis of the MCTL loop equations indicate that a tracking curve (S-curve) is formed from a combination of operating curves based on the direct path and reflected path signal parameters, the sum of which generates a steady-state tracking point of $\delta^* = 0$ when the MCEU estimates are exact.

Analysis was conducted on the enhanced MCEU in order to characterize its performance. A SQP search algorithm locates the minimum of the quadratic cost function $V(\alpha)$, maximizing the PDF of the linear statistical measurement model.

5.2 Computer Simulations

Simulations were conducted using Simulink to determine the tracking performance of the eMRDLL in a realistic multipath environment. Five separate simulations were conducted. The first simulation focused on the stand alone noise-free performance of the MCEU (assumes perfect MCTL synchronization). The second simulation was conducted to analyze the noise-free closed loop code phase tracking performance of the eMRDLL versus the tracking performance of the NCDLL with normalized correlator spacings of $\Delta = 1.0$ and $\Delta = 0.1$. The third simulation focused on the stand alone performance of the MCEU in the presence of AWGN (assumes perfect MCTL synchronization). The fourth simulation was conducted to analyze the closed loop code phase tracking performance of the eMRDLL in the presence of AWGN. Finally, simulations were conducted on the stand alone MCEU (assumes perfect MCTL synchronization) in a time varying multipath environment, with and without AWGN.

Note: SNR performance is relative to the Simulink environment (see Section 4.3.7).

5.2.1 Simulation # 1 Results. Simulations of the noise-free stand alone MCEU resulted in accurate estimates of the multipath parameters within a multipath environment. The MCEU's performance is sensitive to the LPFs used in the bank of correlators. The LPFs function as integrators, convert the received signal to baseband (an enhancement provided by the eMRDLL), and attenuate noise. The response of the filters implemented contributed significantly to the accuracy of the estimates provided by the ML SQP algorithm.

5.2.2 Simulation # 2 Results. Simulations of the stand alone MCEU in the presence of AWGN indicate that it performs well. For the models tested, decreased performance in estimation was not evident until the SNR was reduced to 0 dB. Significant

degradation in MCEU performance was not realized until the SNR was reduced to the identified threshold of $-23\ dB$.

5.2.3 Simulation # 3 Results. Simulations indicate that a bias is generated which progressively hinders the performance of the MCTL over time. Although the bias is present (likely due to nonidealities of the Simulink implementation), code phase tracking of the eMRDLL is still typically better than code phase tracking of the baseline narrow correlator NCDLL with for $\alpha \in [0.1, 1.0]$, with the exception of delays $\alpha = 0.5$, $\alpha = 0.6$, and $\alpha = 0.7$. It is speculated that successful removal of the bias would enhance performance of the eMRDLL such that it would outperform the NCDLL ($\Delta = 0.1$) for all delays $\alpha \in [0.1, 1.0]$. The narrow correlator NCDLL exhibited slightly better performance than the eMRDLL for delays $\alpha \in [1.1, 1.5]$; consequently, code tracking performance of the eMRDLL for these delays could possibly be improved by reducing the correlator spacing within the MCTL to $\Delta = 0.1$. The eMRDLL outperforms the NCDLL with correlator spacing $\Delta = 1.0$ for all delays $\alpha \in [0.1, 1.5]$.

5.2.4 Simulation # 4 Results. Simulations of the eMRDLL in the presence of AWGN indicate that, as with the MCEU, significant degredation in code tracking performance does not occur until the SNR is reduced to $-23\ dB$. Furthermore, small levels of noise, $SNR \approx 0\ dB$, enhanced the performance of the eMRDLL relative to the noiseless case by exciting/perturbing the system to counter the effects of the bias. The tracking quality had a tendency to degrade over time, due to the cummulative effect of the bias.

5.2.5 Simulation # 5 Results. Simulations conducted on the MCEU in a time varying multipath environment indicate that the MCEU provides estimates which are very accurate and comparable with those for the cases of fixed multipath delays. There is typically some estimation error introduced by the signal delay caused by the response of the LPF. As with previous simulations, the MCEU showed no degredation in estimates for SNR$\geq 0\ dB$. Significant degredation in multipath parameter estimation occurred for SNR$\leq -23\ dB$.

5.3 Summary and Recommendations

This thesis has shown that the eMRDLL has tremendous potential for multipath mitigation in GPS applications. Although the parameters used in the simulations were different than those found in GPS, the code tracking loop models exhibited properties that were very similar to the properties encountered in the GPS environment. The eMRDLL exhibited significant improvements in steady state code phase tracking in comparison with the code tracking performance of the standard NCDLL in the presence of a single reflected signal. Furthermore, the eMRDLL exhibited improved code phase tracking in comparison with the code tracking performance of the narrow correlator NCDLL for most delays $\alpha \in [0.1, 1.0]$. Removal of the simulation bias generated by the Simulink implementation should further increase performance. The MLE implemented in the eMRDLL has been shown to accurately estimate the multipath signal parameters for delays $\alpha \in [0.2, 1.5]$. The following is a list of recommendations for follow-on research to further improve the performance of the eMRDLL.

- Reduce the correlator spacing within the eMRDLL to improve code phase tracking performance for delays $\alpha \in [1.05, 1.5]$.

- Generalize the eMRDLL for operation in a multipath environment where multiple reflections are present.

- Examine the effects of Doppler on the code phase of the received multipath signal with and without the ALC implemented.

- Examine the performance of the eMRDLL when random data modulation is included on the received GPS signal, and perform bit error analysis.

- Design a detection/estimation scheme that detects the presence of multipath. This would allow the NCDLL to be used when multipath is not present, since it has been shown to perform better in the absence of multipath [6].

Appendix A. Computer Simulation Models

A.1 Overview

Two basic models were designed using Simulink: the eMRDLL and the NCDLL. Parameters used for the simulations are provided in Section 4.3.8. This appendix provides a description of the Simulink models designed.

A.2 The Transmitted Multipath Signal

The multipath environment was simulated by creating a PN code generator which operated at a rate governed by a pulse generator ($1/T_c$=100 Hz) as shown in Figure 67.

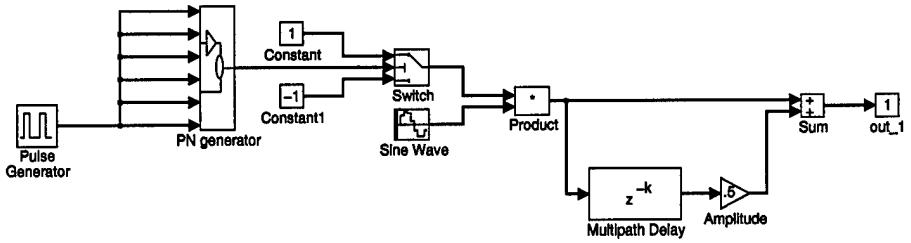


Figure 67 Simulink multipath model.

The PN generator consists of a six bit feedback register, shown in Figure 68, with a polynomial $X^6+X^5+X^3+X^2+1$ of order $m = 6$. This register produces a maximum lenth PN sequence of $N = 2^m - 1 = 63$ [3]. Although GPS transmits a signal with code length $N = 1023$, choosing $N = 63$ for the simulations still permitted the use of the large N code correlation approximation of Equation 8 with the added benefit of increasing simulation performance.

Each register of the PN generator, shown in Figure 69, consists of switches and delays, which are controlled by the pulse rate of the PN generator. Each register is initiated with initial setting equal to one. This is accomplished by setting the initial condition of the unit delay block in each register equal to one. The received pulses from the pulse generator arrive at input 2, which flips the switch allowing the next bit to enter through input 1.

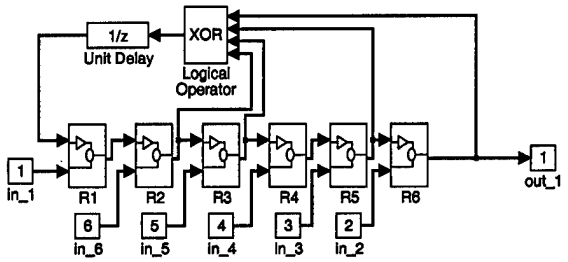


Figure 68 Simulink PN generator model.

This value then flips switch 1 to the corresponding constant value which departs through output 1 and also cycles through the loop until the next pulse flips the switch again. As such, the output of each register cycles through the next register upon receiving a pulse from the pulse generator.

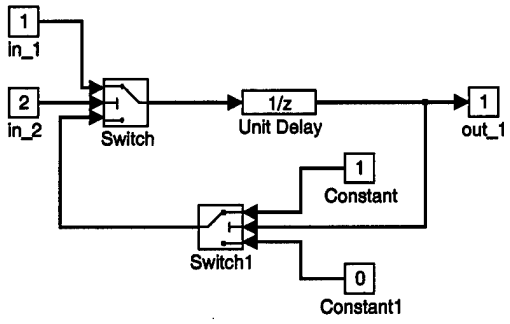


Figure 69 Simulink model of a single register, R, of the PN code generator.

The output chip from the PN generator then proceeds to the switch shown in Figure 68 which flips the switch to +1 for code value equal to 1 or -1 for code value equal to 0. Thus generating a sequence of $\pm$1s. This sequence is then modulated onto a carrier f_o=1000 Hz with phase equal to $\pi/2$ (cosine wave). The sequence of $\pm$1s are represented as 180° phase shifts in the carrier signal. This represents the transmitted direct path signal. The reflected signal is then generated by delaying and attenuating the direct path signal and then summing this signal with the direct path signal to form the multipath signal.

A.2.1 NCDLL Model. The simulink model of the NCDLL is based on the NCDLL illustrated in Figure 4. Figure 70 shows the model implemented in Simulink. The figure indicates that the transmitted signal is delayed prior to entering the 'early-late' gate channels of the NCDLL. This delays the transmitted signal relative to the 'early' signal of the NCDLL by $\Delta T_c/2$ and maintains an advance of the transmitted signal relative to the 'late' signal by $\Delta T_c/2$. The remaining operations of the NCDLL are based on the theory of operations described by the equations in Chapter 2 for the NCDLL. Parameters chosen for the model are listed in Section 4.3.8.

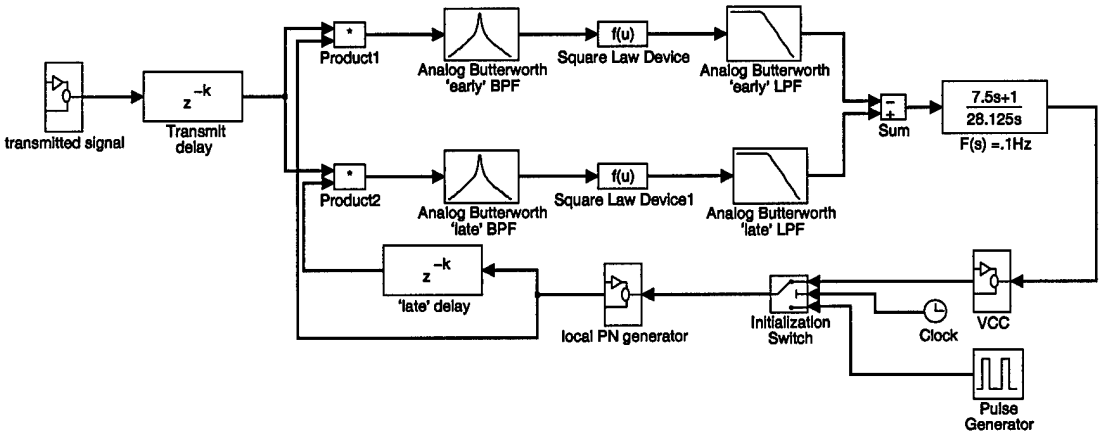


Figure 70 Simulink model of the NCDLL.

A.2.2 Simulink Models of the VCC and the Local PN Generator. The VCC implemented in the NCDLL is shown in Figure 71. The error signal from the output of the loop filter enters the VCC and is multiplied by a gain of -1. This is necessary in order for proper synchronization to occur. The signal then enters the VCO (block provided by the Communications Toolbox) which operates a given quiescent frequency (100 *Hz*). The frequency of the VCO is adjusted according to the input error signal. The sinusoidal output of the VCO then enters a rising edge detector (block provided by

the Communications Toolbox) which produces a pulse at the first positive leading edge of the output of the VCO. The rate of these pulses determines the code rate of the local PN generator shown in Figure 72. Note that the local PN generator operates the same way as the transmitted signal generator previously described. If the received signal is delayed (out of synchronization) with the locally generated signal then the discriminator output will be some positive voltage level as described by the S-curve for the NCDLL. After passing through the loop filter, this signal is multiplied by a negative gain of one and passed through to the VCO. The negative signal input to the VCO reduces the frequency of the sinusoidal outputs of the VCO which reduces the rate of the clock pulses produced by the rising edge detector. As such, the rate of the local PN generator is reduced to match the delayed received signal.

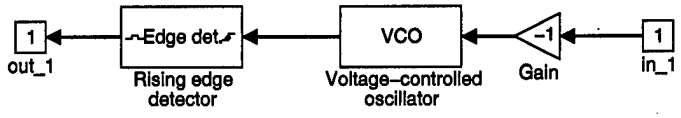


Figure 71 Simulink model of the VCC.

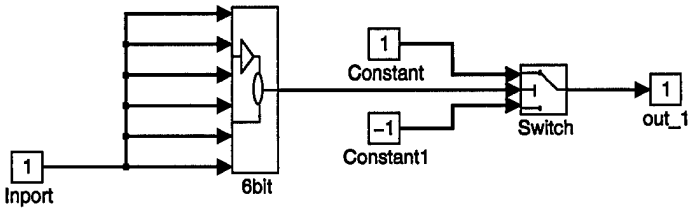


Figure 72 Simulink local PN generator model.

A.3 eMRDLL

The simulink model of the eMRDLL, illustrated in Figure 73 is based on the eMRDLL illustrated in Figure 15. The basic components of the Simulink model include the

- transmitted signal and transmission delay (previously discussed),
- AWGN function block (block provided by the Communications Toolbox),
- PLL model,
- direct and reflect path 'early-late' gate models,
- ALC model,
- MCEU model,
- variable delay function block,
- VCC (previously discussed),
- local PN generator model (previoulsy discussed).

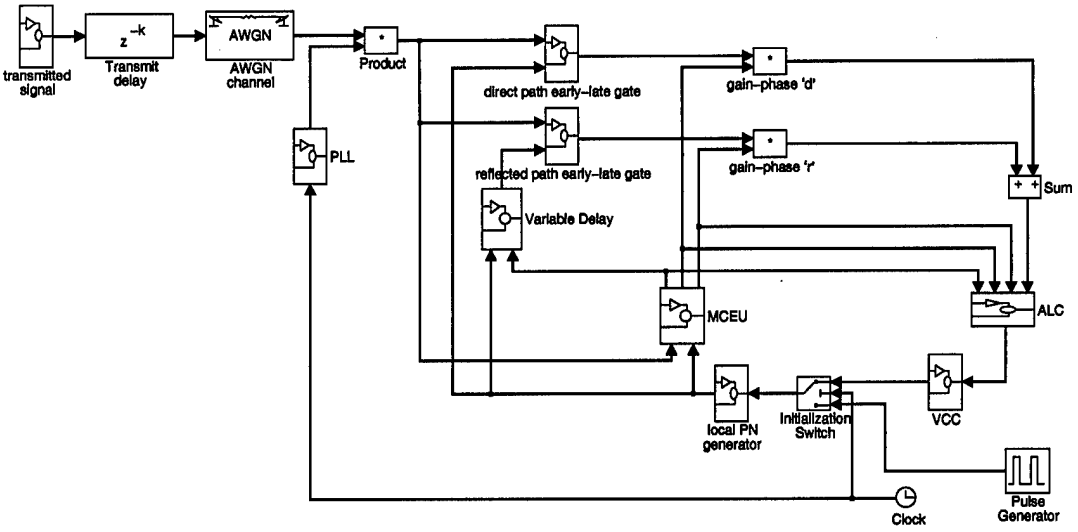


Figure 73 Simulink model of the eMRDLL.

A.3.1 Simulink AWGN Block. The AWGN block is a Simulink function block provided by the Communications Toolbox. There are three parameter settings for this

block: the mean, the seed value, and the variance. The mean was always set to zero. The seeds were varied for different SNRs as specified in Table 7. The variance of the noise was used in determining the SNR as explained in Section 4.3.7.

Table 7 Specified SNR with corresponding seed value.

Specified SNR (dB)	Seed Value (dB)
30	12345
25	12345
20	12345
15	46531796
10	12345
5	46531796
0	46531796
-6	84531796
-13	84531796
-23	84531796
-35	84531796

A.3.2 Simulink PLL Model. The received signal is mixed with a signal from the PLL which is designed as a black box that provides the theoretical PLL signal as described in Chapter 3. The mixed signal then enters the MCEU and the 'early-late' gates of the MCTL.

A.3.3 Simulink 'Early-Late' Gate Model. The 'early-late' gate design of the direct and reflected paths is illustrated in Figure 74. This is a typical coherent design in which the received signal is converted to baseband and mixed with a locally generated code replica spaced at $\pm\Delta T_c/2$ in the 'early' and 'late' channels. Low pass filtering the signals and differencing provides the discriminator in each channel as defined in Chapter 3. The signals pass through the *gain-phase correlators* and are summed to form the eMRDLL discriminator output (S-curve).

A.3.4 Simulink ALC Model. The discriminator output signal enters the ALC illustrated in Figure 75 which is based on the Figure 22 and Equations 92 and 93. $K_A = 1/\tau_1 = 8/225\hat{A}$ is determined by Equation 94 where $\omega_n \approx 0.06\pi\ rad/sec$, $K_\circ = 1$, and

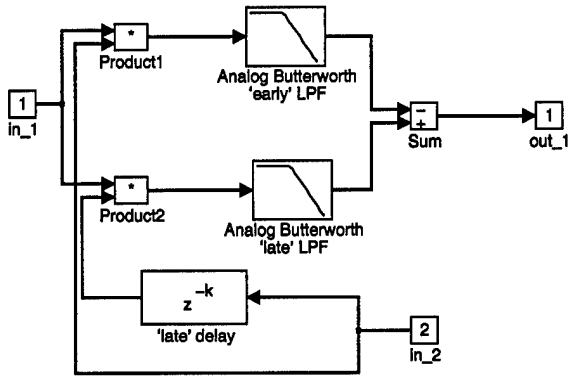


Figure 74 Simulink 'early-late' gate model used in the eMRDLL.

$1/\hat{A}$ is determined by the matlab function *alc.m* based on the inputs from the MCEU. $\tau_2 = 2\zeta/\omega_n = 7.5$ where $\zeta = 1/\sqrt{2}$ remained fixed as with the NCDLL. The output of the ALC drives the VCC which drives the local PN generator as previously described for the NCDLL.

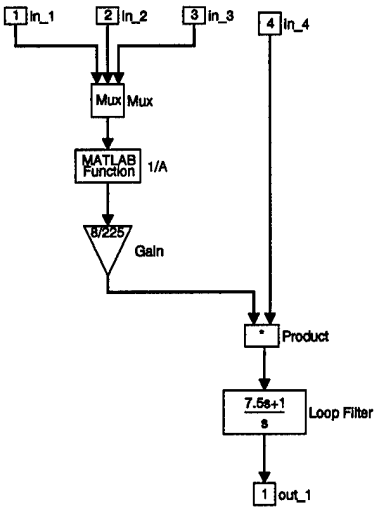


Figure 75 Simulink model of the ALC.

A.3.5 Simulink MCEU Model. The mixed signal entering the MCEU is converted to baseband and correlated with locally generated PN code incrementally spaced by $B_k T_c$. The MCEU implemented in Simulink is illustrated in Figure 76. Two inputs exist into

the MCEU. The received (mixed) signal enters the MCEU through input 1. The locally generated spreading code enters the MCEU through input 2. Note that the locally generated code is initially delayed by the same delay as the transmitted delay. This ensures that the locally generated code is synchronized with the transmitted code. The locally generated code then enters a bank of correlator delays which are incrementally spaced by 0.15 through the range $\beta_k \in [0, 1.5]$. the delayed code then enters the bank of correlators were it is mixed/multiplied with the received (mixed) signal and low pass filtered. The LPFs serve to convert the signal to baseband and also operate as integrators. The corre-

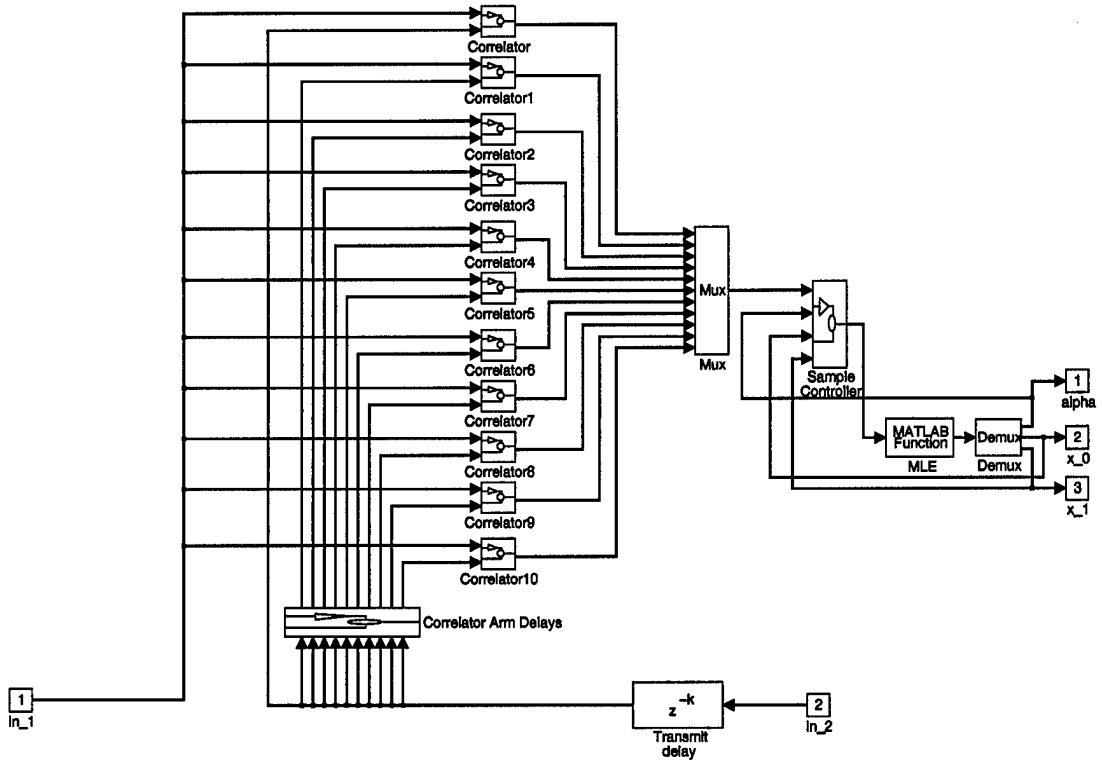


Figure 76 Simulink model of the MCEU.

lator samples, $\mathbf{R}$, enter the MLE sample controller block shown in Figure 77. The sample controller block is controlled by the pulse generator which is connected to the switch. The pulse generator sends a pulse at the sampling rate chosen for the MLE, $NT_c = 0.63$. The switch, when triggered by the pulse generator, allows the correlator samples $\mathbf{R}$ to pass through at the appropriate sampling epoch. Otherwise, the switch passes inputs from the

mux which contains three memory blocks for $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$. These memory blocks contain feedback from the MLE. The general concept of the memory block is to pass new correlator samples to the estimator every estimator sampling period, NT_c. Otherwise, the previous estimates of $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ are passed through the MLE. The MLE, which is written as a Matlab function $(mle.m)$[1], is designed to calculate new estimates $\hat{\alpha}$, $\hat{x}_0$, and $\hat{x}_1$ when new correlator samples are received and pass through previous unchanged estimates, which helps to reduce processing time. The MLE calculates estimates by using an SQP algorithm as described in Chapter 3.

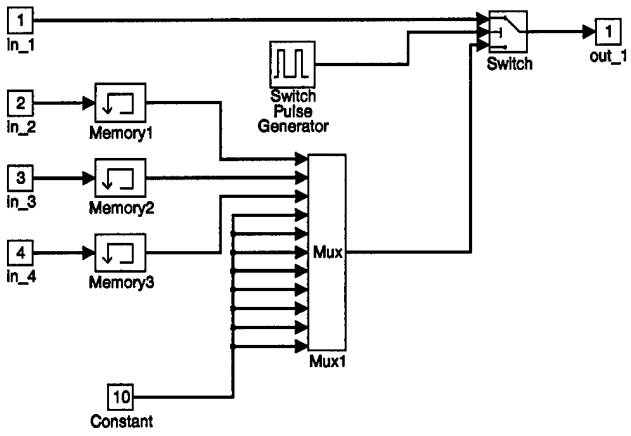


Figure 77 Simulink model of the sample controller block.

Estimates from the MCEU are provided to three different functional blocks within the eMRDLL: the ALC, the gain-phase multipliers, and the variable delay function block.

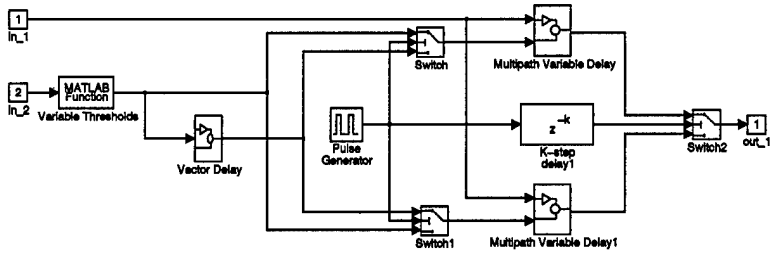


Figure 78 Simulink model of the variable delay function block.

[1]This function requires the Optimization Toolbox.

A.3.6 Simulink Variable Delay Model. The variable delay function block, shown in Figure 78, controls the delay of spreading code from the local PN generator to ensure proper synchronization within the MCTL reflected path channel. The locally generated spreading code enters the variable delay function block through input 1 and is input into two multipath variable delay blocks. Each of the two multipath variable delay blocks, shown in Figure 79, is comprised of the same components. These components consist of several switches of which any combination of delay $\alpha \in (0, 1.5]$ in increments of .01 can be formed. The individual delay blocks are controlled by switches. These switches are controlled by the variable thresholds block located within the variable delay function block. The variable thresholds block is a Matlab function (*vdelay.m*) which controls the combination of switches in the multipath delay block based on the estimated delay $\hat{\alpha}$ provided by the MCEU.

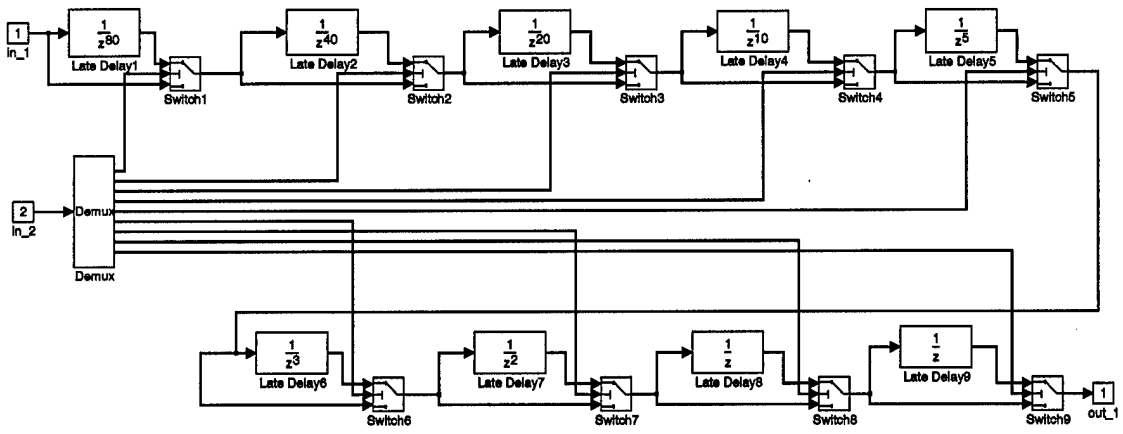


Figure 79 Simulink model of the multipath variable delay block.

The output of the two multipath delay blocks are controlled by a pulse generator connected to a switch which is set such that outputs from the two blocks are alternately passed through the switch every other estimator sample cycle NT_c. Two separate multipath delay blocks are used to ensure that all delayed chip samples within the combination of delay blocks of the multipath variable delay block are properly cleared when a new estimate of α is provided by the MCEU.

A.4 Matlab Functions Used in Simulink Models

The following Matlab m-files are functions used in the Simulink components of the eMRDLL.

A.4.1 mpll.m.

```
function signal= mpll(parameters)
% MPLL is a function which produces a signal that models the PLL for MRDLL.
% The input parameters provides t, alpha, a0, and a1.
% t is the time instant. alpha is the multipath delay. a0 is the amplitude of
% the direct path signal. a1 is the amplitude of the reflected signal.

% Authored by Fred Baier
% Created 20 Oct 97

t=parameters(1);
alpha=parameters(2);
a0=parameters(3);
a1=parameters(4);

f_c=1000;
T_c=1/100;
theta0=0;

if alpha <= 1
  R_c=1-alpha;
else
  R_c=0;
end
```

A.4.2 alc.m.

```
function Gain=alc(estimates)
% ALC is a function that determines the gain of the Adaptive Loop Controller
% in the Simulink environment based on the estimates of alpha, x0, and x1.

% Authored by Fred Baier
% Created:  17 Sep 97
% Modified:  9 Oct 97

alpha=estimates(1);
x0=estimates(2);
x1=estimates(3);

if alpha <= 0.5
  A = 2*(x0^2+x1^2+2*x0*x1);
elseif alpha == 0.5
  A = 2*(x0^2+x1^2+.5*x0*x1);
elseif alpha == 1.5
  A = 2*(x0^2+x1^2-.5*x0*x1);
else
  A = 2*(x0^2+x1^2-x0*x1);
end

Gain=1/A;
```

A.4.3 mle.m.

```
function output=mle(R_in)
%function [alpha,x_0,x_1]=mle(R)

% MLE
% Authored by Fred Baier
```

```
% Functions required:  crosscor.m, toeplitz, constr.m, sig1_H.m, V_alpha.m

% Function created for simulink:  25 Aug 97
% Last modified:  3 Sep 97, 22 Oct 97

if sum(R_in(4:11))==80
  output=R_in(1:3);  % NOTE:  Output will be a vector
else

% THE FOLLOWING CALCULATES NEW OUTPUT VECTOR
global R Beta C M;
R=R_in;

% Create noise covariance matrix, C.

M=11;  \% number of arms in MRDLL
increment=0.15;  % delay spacing between arms

Beta=[0:increment:1.5]';
Beta_diff=Beta;

R_c=crosscor(Beta_diff,M);
C=toeplitz(R_c);

% STEP 1 is performed by function V_alpha.
% global R Beta C M;

% STEP 2.  Invoke an optimization routine.
f_old=1e6; % Start with large number as placeholder
alpha_old=2;
```

```
for arm=1:10
  if arm==1
    start_alpha=.1;
    VLLB=.05;
    VLUB=.14999;
    %VLLB=.06;
    %VLUB=.149;
  elseif arm==2
    start_alpha=.2;
    VLLB=.15001;
    VLUB=.29999;
    %VLLB=.151;
    %VLUB=.299;
  end
  junk=foptions;
  alpha=constr('V_alpha',start_alpha,junk,VLLB,VLUB);
  H_min=sig1_H(Beta,alpha,M);
  f=(inv(C)*R_in)'*(C-H_min*inv(H_min'*inv(C)*H_min)*H_min')*(inv(C)*R_in);

  if f_old < f
    alpha=alpha_old;
  else
    f_old=f;
    alpha_old=alpha;
  end
%alpha  % ADDED TO TRACK IN SIMULINK

  start_alpha=start_alpha+.15;
  VLLB=VLLB+.15;
```

```
  VLUB=VLUB+.15;
end

R_in;  % FOR TRACKING IN SIMULINK
alpha;
H=sig1_H(Beta,alpha,M);
x_est=inv(H'*inv(C)*H)*H'*inv(C)*R_in;
x_0=x_est(1);
x_1=x_est(2);

output=[alpha x_0 x_1]';

end
```

A.4.3.1 crosscor.m.

```
function R_c=crosscor(Omega,M)
% CROSSCOR produces cross correlation for input Omega
for index_a=1:M
  if abs(Omega(index_a,1)) <= 1
    R_c(index_a,1)=1-abs(Omega(index_a,1));
  else
    R_c(index_a,1)=0;
  end
end
```

A.4.3.2 sig1_H.m.

```
function H=sig1_H(Beta,alpha,M)
% SIG1_H produces the regressor matrix, H, of signal for vector input Beta, (a
%   vector of delays for each arm), alpha (delay of the reflected signal, single
%   reflection case), and M (the number of arms in the DLL). Requires
```

```
%   the use of the function file CROSSCOR.
%   Authored by Fred Baier.

Omega_1=Beta;
R_c1=crosscor(Omega_1,M);
Omega_2=alpha-Beta;
R_c2=crosscor(Omega_2,M);
H=[R_c1 R_c2];
```

A.4.3.3 V_alpha.m.

```
function [f,g]=V_alpha(alpha)
% V_ALPHA is a function which determines alpha by minimizing the MLE equ.
%   Want to minimize MLE with constraints on alpha.
%   R is correlator output data (known). Beta is the delay of the arms of
%   MRDLL. alpha is the multipath delay to be estimated. C is the noise
%   covariance matrix.

global R Beta C M;

% STEP 1.
%f=V_a. This is the function that we want to be minimized.
H=sig1_H(Beta,alpha,M);
f=(inv(C)*R)'*(C-H*inv(H'*inv(C)*H)*H')*(inv(C)*R);

g(1)=-alpha;
g(2)=alpha-1.5;
```

Bibliography

1. Best, Roland E. *Phase-locked Loops*. New York, New York: McGraw-Hill,Inc., 1984.
2. Gardner, Floyd M. *Phaselock Techniques*. New York, New York: John Wiley & Sons, 1979.
3. Jeruchim, Michel C. and others. *Digital Communications Fundamentals and Applications*. New York, New York: Plenum Press, 1992.
4. J.J. Spilker, Jr. "GPS Signal Structure and Performance Characteristics," *Global Positioning System: Papers Published in NAVIGATION*, *1*:29–54 (1980).
5. Kay, Steven M. *Fundamentals of Statistical Signal Processing Estimation Theory*. Upper Saddle River, New Jersey: Prentice-Hall, 1993.
6. Laxton, Captain Mark C. *Analysis and Simulation of a New Code Tracking Loop for GPS Multipath Mitigation*. MS thesis, AFIT/GE/ENG/96D-13, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, December 1996.
7. Laxton, Mark C. and Stewart L. DeVilbiss. "GPS Multipath Mitigation During Code Tracking." *Proceedings of the American Control Conference*. 1429–1433. June 1997.
8. Parkinson, Bradford W. and others. *Global Positioning System: Theory and Applications, Volume 1*. Cambridge, Massachusetts: American Institute of Aeronautics and Astronautics, Inc., 1996.
9. Peterson, Roger L. and others. *Introduction to Spread Spectrum Communications*. Englewood Cliffs, New Jersey: Prentice-Hall, 1995.
10. Sheen, Wern-Ho and Gordon L. Stuber. "A New Tracking Loop for Direct Sequence Spread Spectrum Systems on Frequency Selective Fading Channels." *Proceedings of the IEEE International Conference on Communications*. 1364–1368. June 1995.
11. Simon, Marvin K. "Noncoherent Pseudonoise Code Tracking Performance of Spread Spectrum Receivers," *IEEE Transactions on Communications*, *25*(3):327–345 (March 1977).
12. Spilker, Jr., James J. *Digital Communications by Satellite*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1977.
13. Stuber, Gordon L. and others. "A Coherent Tracking Loop for Direct Sequence CDMA Systems." *First European Personal and Mobile Communications Conference*. 221–226. November 1995.
14. Townshend, Brian R. and others. "Performance Evaluation of the Multipath Estimating Delay Lock Loop," *NAVIGATION: Journal of the Institute of Navigation*, *42*(3):503–514 (Fall 1995).
15. van Nee, D.J.R. "Reducing Multipath Tracking Errors in Spread-Spectrum Ranging Systems," *Electronics Letters*, *28*(8):29–54 (April 1992).

16. van Nee, Richard D.J. "The Multipath Estimating Delay Lock Loop." *Proceedings of the IEEE Second International Symposium on Spread Spectrum Techniques and Applications*. 39–42. November 1992.

17. van Nee, Richard D.J. and Jaap Siereveld. "The Multipath Estimating Delay Lock Loop: Approaching Theoretical Accuracy Limits." *Proceedings of the IEEE Position, Location and Navigation Symposium*. 246–251. April 1994.

18. VanDierendonck, A.J. and others. "Theory and performance of narrow correlator spacing in a GPS receiver," *NAVIGATION: Journal of the Institute of Navigation*, *39*(3):265–283 (Fall 1992).

19. VanNee, Richard. "Multipath effects on GPS code phase measurements," *NAVIGATION: Journal of the Institute of Navigation*, *39*(2):177–190 (Summer 1992).

20. Weill, Lawrence. "C/A Code Pseudoranging Accuracy - How Good Can It Get?." *Proceedings of the ION GPS-94 7th International Technical Meeting*. 133–141. September 1994.

21. Weill, Lawrence R. "Conquering Multipath: The GPS Accuracy Battle," *GPS WORLD*, *8*(4):59–66 (April 1997).

22. Zahirniak, Major Daniel R. *Parameter Estimation For Real, Filtered Sinusoids*. PhD dissertation, AFIT/DS/ENG/96-06, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, September 1997.

23. Ziemer, R.E. and W.H. Tranter. *Principles of Communications*. New York: John Wiley and Sons, Inc., 1995.

Vita

Captain Fred P. Baier was born in Crossville, Tennessee on June 23, 1970. He graduated from Cumberland County High School in 1988. Following his graduation, he attended The Georgia Institute of Technology where he graduated with the degree Bachelor of Electrical Engineering and was commissioned as a Second Lieutenant in the U.S. Air Force on September 3, 1992. He entered active duty on March 15, 1993 arriving at Hill AFB, Utah. At Hill, he worked as the F-15 Program Engineer for the Training Systems Management Division for three years. In May of 1996, he was assigned to the Air Force Institute of Technology to earn his Master of Science Degree in Electrical Engineering.

Permanent address: 978 Baier Road
Crossville, TN 38555

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1997	Master's Thesis

4. TITLE AND SUBTITLE

A GPS CODE TRACKING RECEIVER DESIGN FOR MULTIPATH MITIGATION USING MAXIMUM LIKELIHOOD ESTIMATION

5. FUNDING NUMBERS

6. AUTHOR(S)

Fred P. Baier

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GE/ENG/97D-18

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFIWC/SAA
2241 Avionics Circle, Bldg 620
WPAFB, OH 45433-7333

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT

Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The NAVSTAR Global Positioning System (GPS) is currently used in many applications requiring precise positioning data. Improving the precise positioning information requires the removal of errors that perturb the received signals. The errors introduced by multiple propagation channels, termed multipath, are not easily removed. These channels are caused by reflective surfaces near the receiver. As such, multipath is uncorrelated between receivers and, thus, cannot be removed through differencing techniques.

This thesis investigates a GPS code tracking loop design which uses maximum likelihood (ML) estimation to determine amplitude and phase information of the multipath signal which are used to adjust code tracking to account for multipath effects. Analysis of the operations that govern this design for the case of a single reflection shows that it has no steady state tracking error. Results of simulations indicate that the code tracking loop, in conjunction with the MLE, mitigate the effects of multipath and improves code tracking performance over the narrow correlator NCDLL for most scenarios analyzed. Overall results of simulations indicate that the implementation of the maximum likelihood estimator (MLE) in conjunction with the code tracking loop has the potential to enhance code tracking performance over that offered by the narrow correlator NCDLL in a GPS environment.

14. SUBJECT TERMS	15. NUMBER OF PAGES
GPS, Global Positioning System, Multipath, Maximum Likelihood Estimation	135
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL